Support Vector Machines

전 재 욱

Embedded System 연구실
성균관대학교

Embedded System Lab.

Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

Embedded System Lab.

Supervised Learning Algorithms

- A range of different algorithms
  - With supervised learning algorithms - performance is pretty similar
    - What matters more often is;
      - The amount of training data
      - Skill of applying algorithms

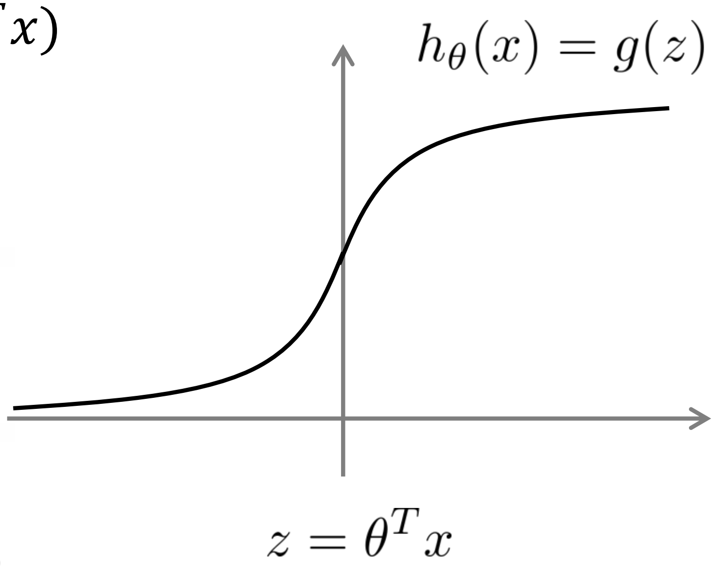
- Other supervised learning algorithm widely used
  - Support vector machine (SVM)
    - Compared to both logistic regression and neural networks,

    a SVM sometimes gives a cleaner way of learning non-linear functions

Logistic Regression

- Logistic regression

$$h_\theta(x) = \frac{1}{1 + \exp(-\theta^T x)}$$



  - If $y = 1$, we want $h_\theta(x) \approx 1,\ \theta^T x \gg 0$
  - If $y = 0$, we want $h_\theta(x) \approx 0,\ \theta^T x \ll 0$

Alternative View of Logistic Regression

- Contribution of each example to the overall cost function

$$y^{(i)}\left(-\log h_\theta(x^{(i)})\right) + (1 - y^{(i)})\left(-\log(1 - h_\theta\left(x^{(i)}\right)\right)$$
$$= -y^{(i)} \log \frac{1}{1 + \exp(-\theta^T x^{(i)})} - \left(1 - y^{(i)}\right) \log\left(1 - \frac{1}{1 + \exp(-\theta^T x^{(i)})}\right)$$

- Overall cost function

$$\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\left(-\log h_\theta(x^{(i)})\right) + (1 - y^{(i)})\left(-\log(1 - h_\theta\left(x^{(i)}\right)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

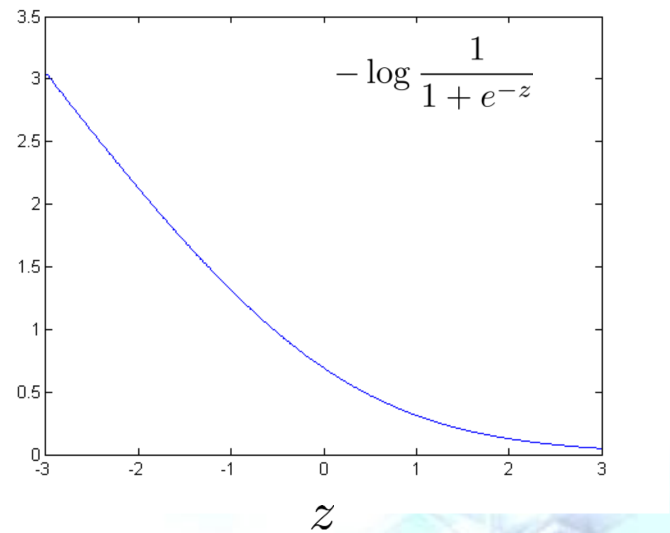
Alternative View of Logistic Regression

- Cost contribution of each example

$$y^{(i)}\left(-\log h_\theta(x^{(i)})\right) + (1-y^{(i)})\left(-\log(1-h_\theta(x^{(i)})\right)$$

$$= -y^{(i)}\log\frac{1}{1+\exp(-\theta^T x^{(i)})} - \left(1-y^{(i)}\right)\log\left(1-\frac{1}{1+\exp(-\theta^T x^{(i)})}\right)$$

- If $y = 1$, (want $\theta^T x \gg 0$)
  - Only $-\log\frac{1}{1+\exp(-\theta^T x)}$ term
  - Cost contribution
    - Low for large $z = \theta^T x$
    - High for small z



➔ This is why when logistic regression sees a positive example($y = 1$), it tries to set $\theta^T x$ to be a very large term ($\theta^T x \gg 0$)

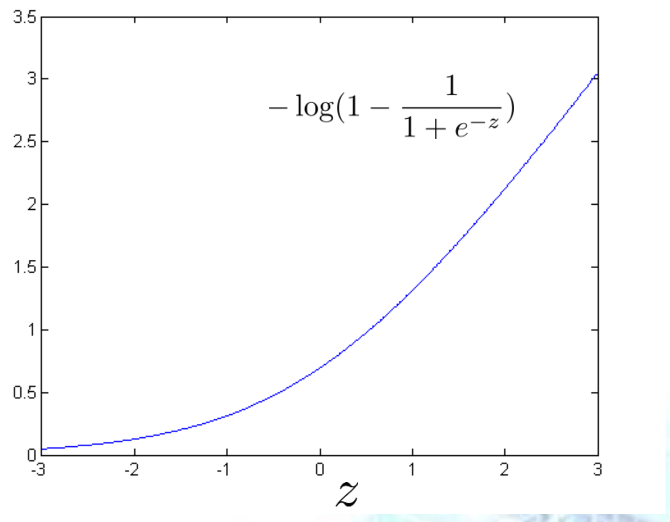
Alternative View of Logistic Regression

- Cost contribution of each example

$$y^{(i)}\left(-\log h_\theta(x^{(i)})\right) + (1-y^{(i)})\left(-\log(1-h_\theta(x^{(i)})\right)$$

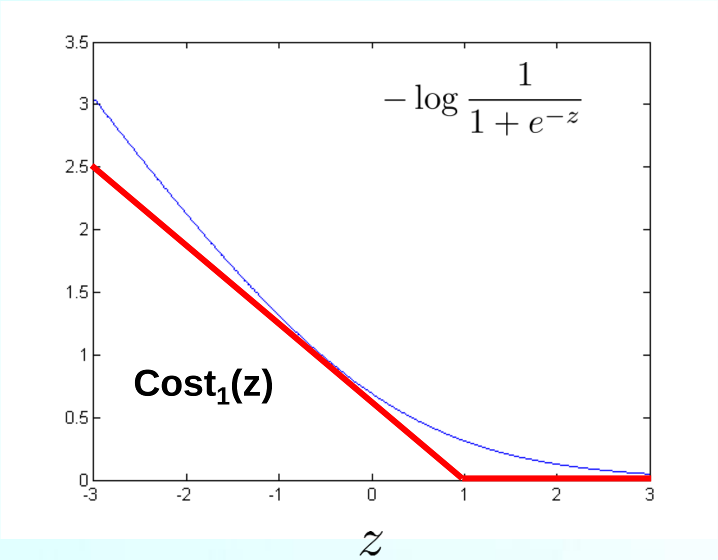
$$= -y^{(i)}\log\frac{1}{1+\exp(-\theta^T x^{(i)})} - \left(1-y^{(i)}\right)\log\left(1-\frac{1}{1+\exp(-\theta^T x^{(i)})}\right)$$

- If $y = 0$, (want $\theta^T x \ll 0$)
  - Only $-\log\left(1-\frac{1}{1+\exp(-\theta^T x)}\right)$ term
  - Cost contribution
    - Low for small z
    - High for large z



SVM Cost Function

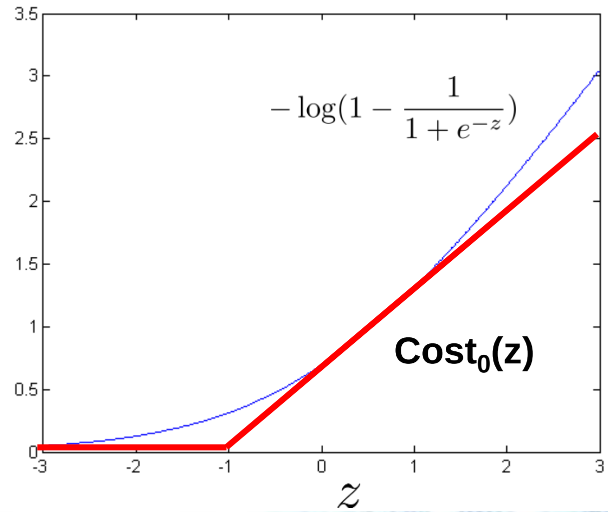
- To build a SVM, redefine our cost functions
  - When $y = 1$
    - Take the ($y = 1$) function and create a new cost function, $\text{Cost}_1(z)$
      - Approximation to the logistic regression ($y = 1$) function
        - two straight lines
  - New ($y = 1$) cost function gives
    - SVM a computational advantage and an easier optimization problem



Embedded System Lab.

SVM Cost Function

- Redefine our cost functions
  - When $y = 0$
    - Take the $(y = 0)$ function and create a new cost function, $\text{Cost}_0(z)$
      - Approximation to the logistic regression $(y = 0)$ function
        - two straight lines
  - New $(y = 0)$ cost function gives
    - SVM a computational advantage and an easier optimization problem



Support Vector Machine

- Logistic regression

$$min_\theta \frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\left(-\log h_\theta(x^{(i)})\right) + (1-y^{(i)})\left(-\log(1-h_\theta\left(x^{(i)}\right))\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Support Vector Machine

- Logistic regression

$$min_\theta \frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\left(-\log h_\theta(x^{(i)})\right) + (1-y^{(i)})\left(-\log(1-h_\theta\left(x^{(i)}\right))\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

- Support vector machine

$$min_\theta \;\; C\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) \; + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

  - Unlike logistic regression, (in SVM) $h_\theta(x)$ does not give us a probability, but instead we get a direct prediction of 1 or 0
    - $h_\theta(x) = \begin{cases} 1 & if \quad \theta^T x \geq 0 \\ 0 & if \quad \theta^T x < 0 \end{cases}$

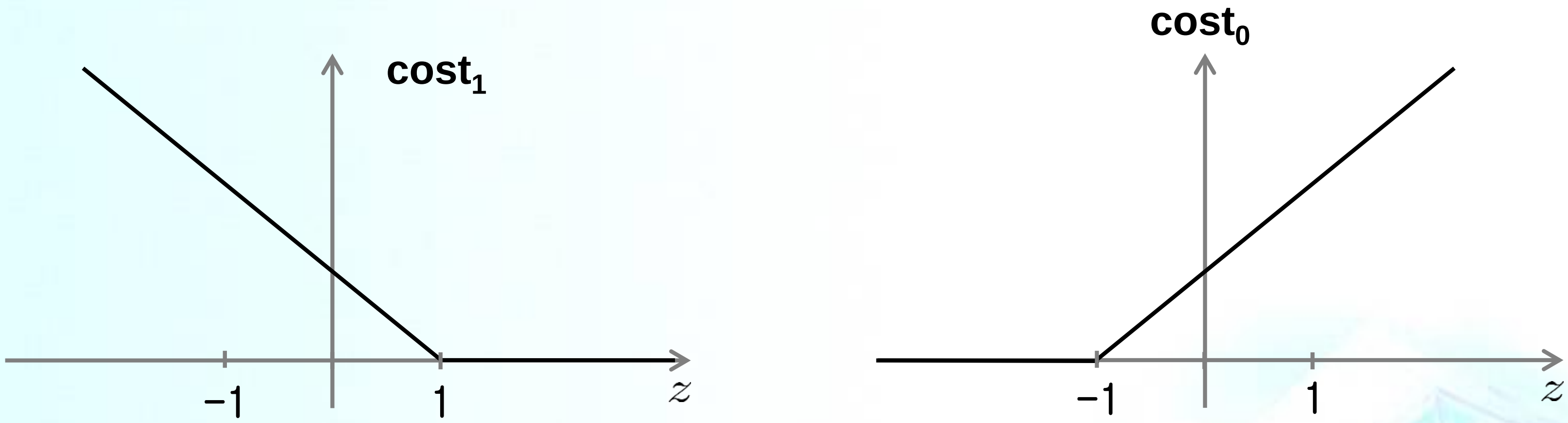
Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

Support Vector Machine

- Support vector machine
  - Large margin classifiers

$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$



If $y = 1$, we want $\theta^T x \geq 1$ (not just ≥ 0)

If $y = 0$, we want $\theta^T x \leq -1$ (not just < 0)

SVM Property

- If we have a positive example,
  - we only really *need* $\theta^T x \geq 0$
    - If $\theta^T x \geq 0$, then we predict 1

- SVM wants a bit more than that
  - It does not want to "just" get it right,

    but have the value be quite a bit bigger than zero

  ➔ Throws in an extra safety margin factor

SVM Property

- Consider a case of huge C
  - For example, C= 100,000
  - Considering the minimization of $CA + B$,
    - pick the zero A value for huge C
    - How do we make A = 0?

  - Making $A = 0$
    - If $y = 1$
      - We need to find a value of θ s.t $\theta^T x \geq 1$ in order to make $A = 0$
    - If $y = 0$
      - We need to find a value of θ s.t $\theta^T x \leq -1$ in order to make $A = 0$

Embedded System Lab.

SVM Property

- If we think of our optimization problem min($CA + B$)
  a way to ensure that this first " A" term = 0,
  - we re-factor our optimization problem into just minimizing the " B " (regularization) term, because $A * C = 0$ when $A = 0$

- That is, try to miminize B , under the following constraints

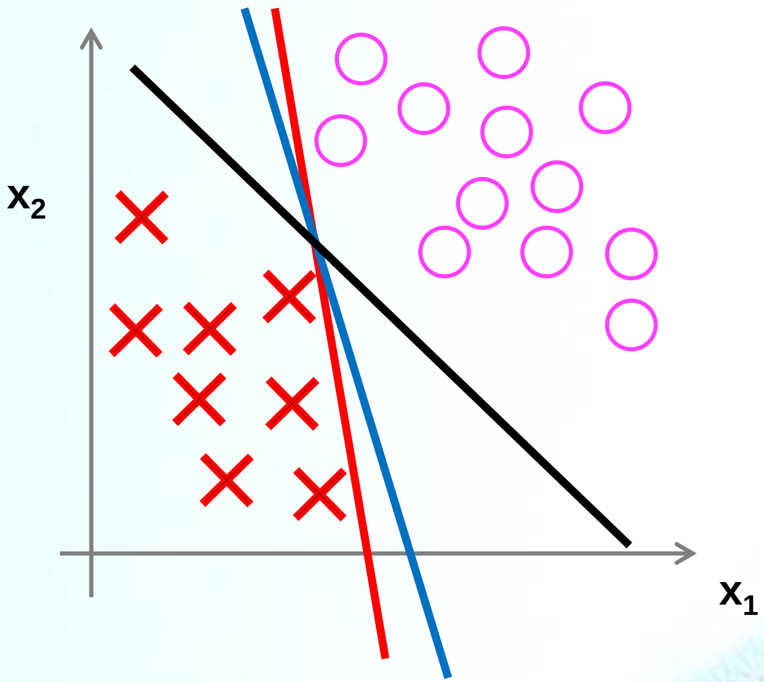
$$min_\theta \left(\frac{1}{2} \sum_{j=1}^{n} \theta_j^2 \right)$$

such that $\theta^T x \geq 1$ if $y^{(i)} = 1$

$\theta^T x \leq -1$ if $y^{(i)} = 0$

Decision Boundaries of Linearly Separable Case

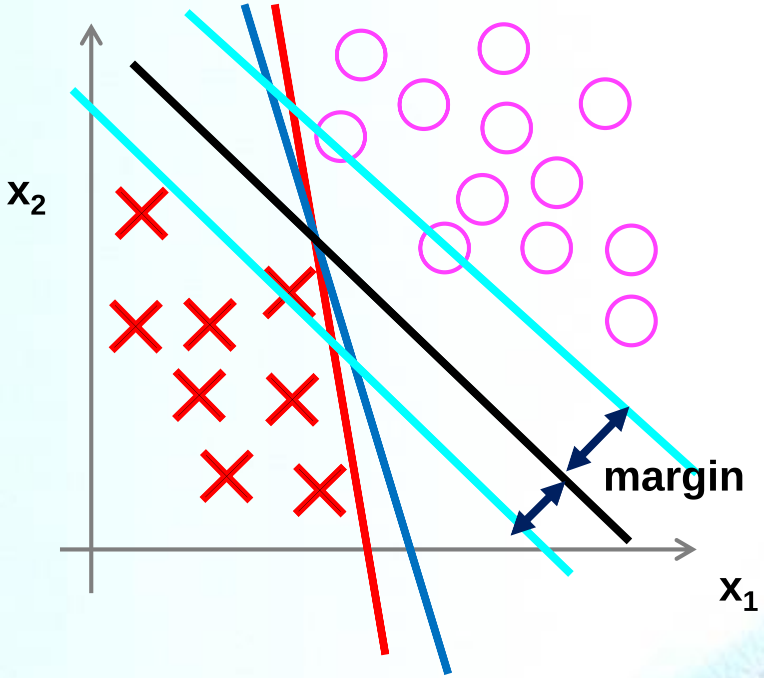
- Decision boundaries of Logistic regression: blue and red
  - They may not generalize too well
- Decision boundary of SVM: black
  - More robust separator



Decision Boundaries

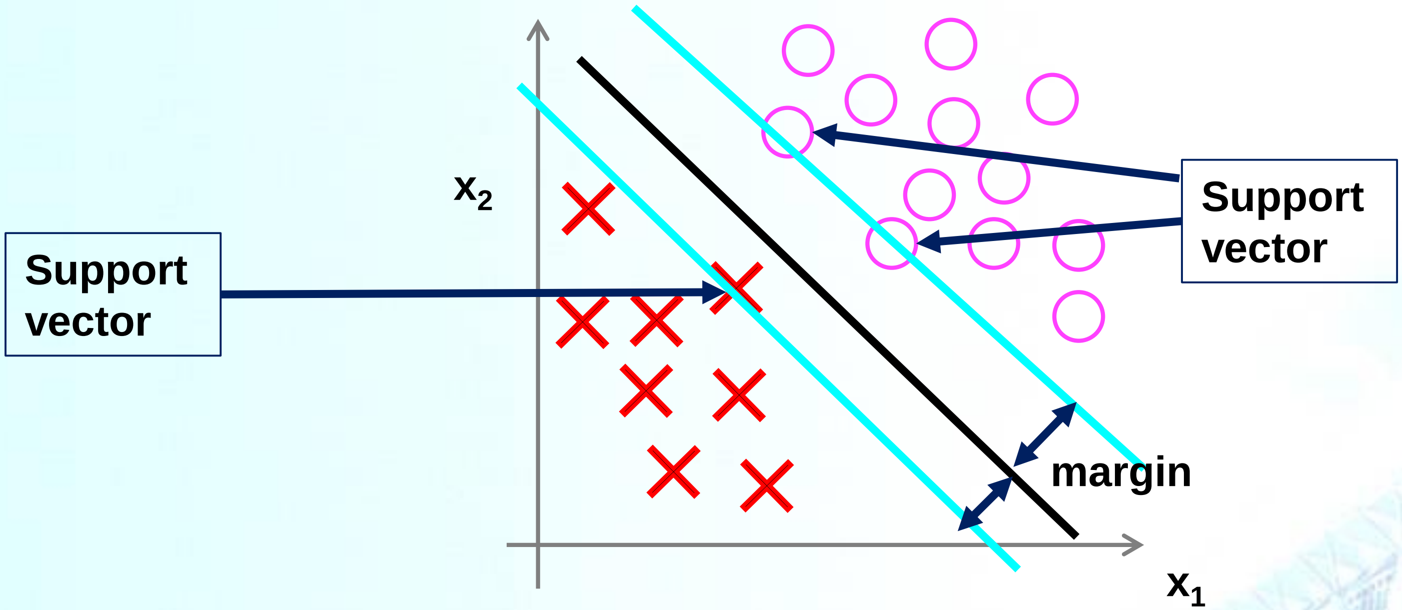
- Decision boundary of SVM
  - More robust separator
    - Mathematically, the black line has a larger minimum distance (margin) from any of the training examples
  - By separating with the largest margin,

    we incorporate robustness into our decision making process



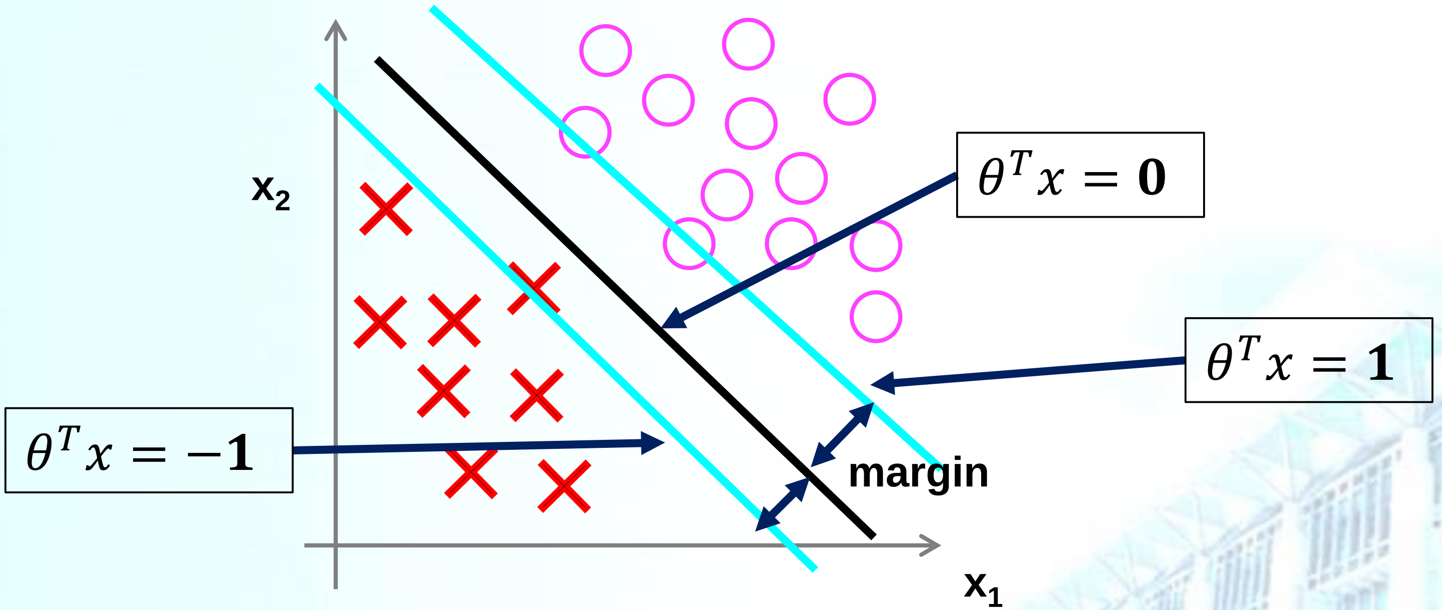
Support Vectors

- Support vectors
  - Data points that lie closest to the decision surface (or hyperplane)
  - They are the data points most difficult to classify
  - They have direct effect on the optimum location of the decision surface



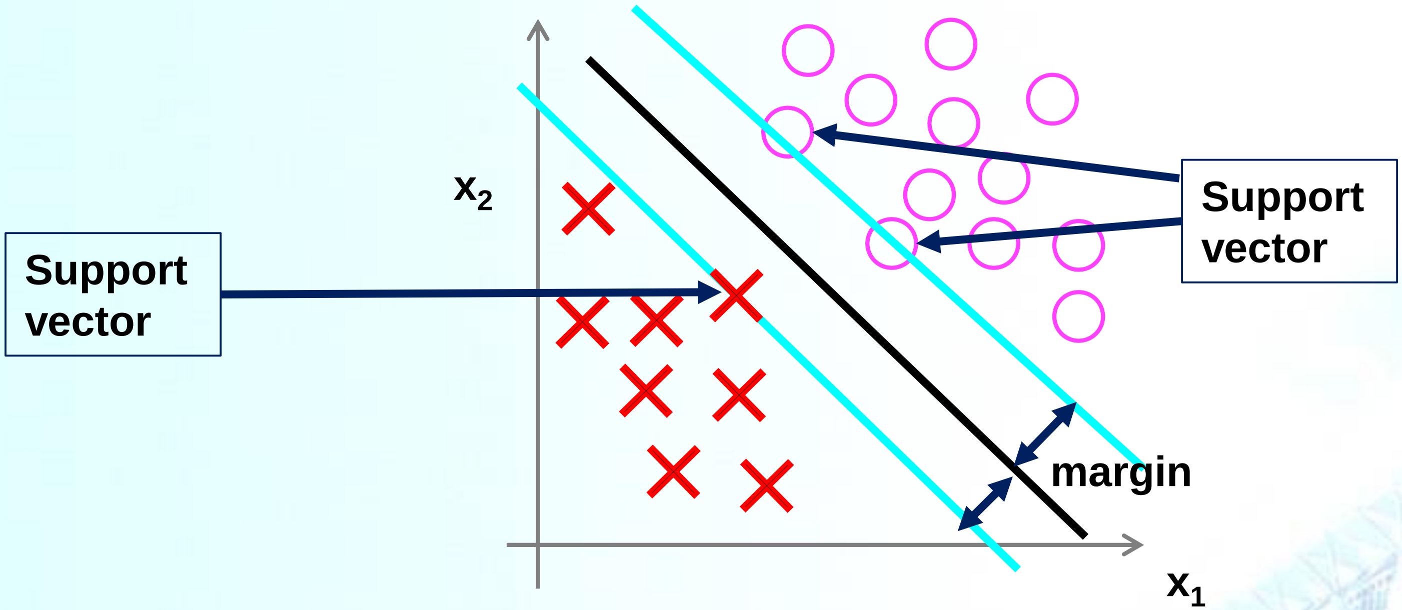
Support Vectors

- The decision function
  - is fully specified by a (usually very small) subset of training samples, the support vectors.



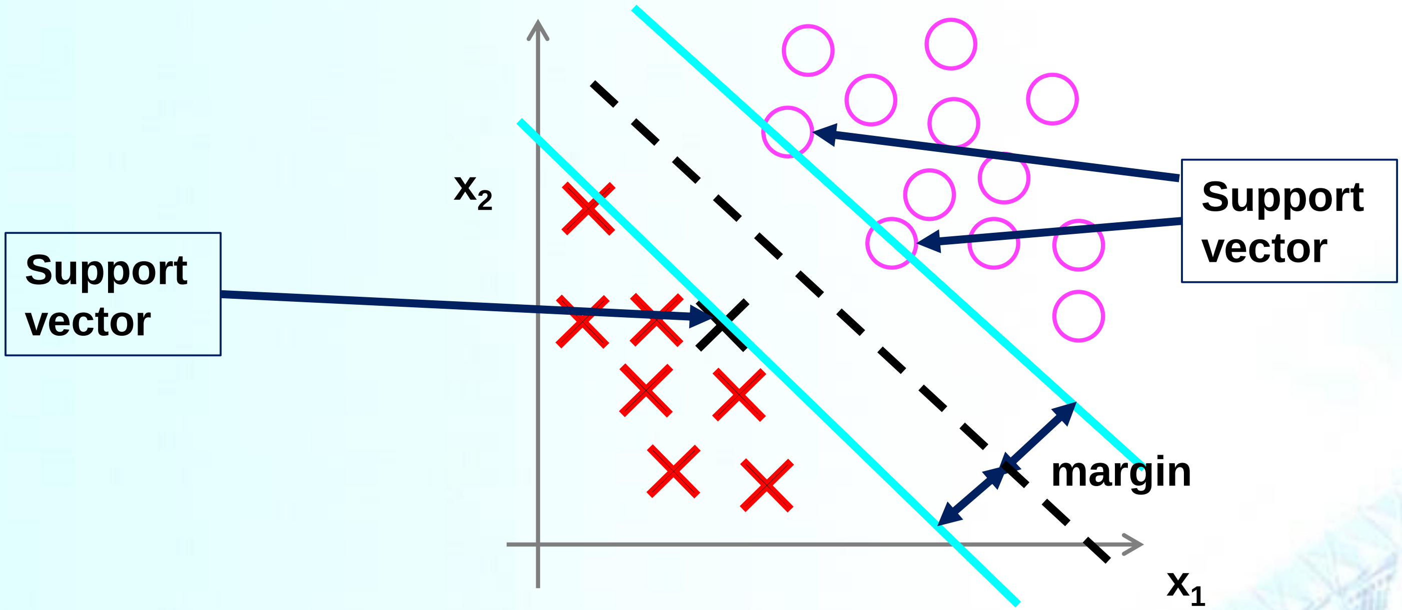
Support Vectors

- Moving a support vector
  - moves the decision boundary



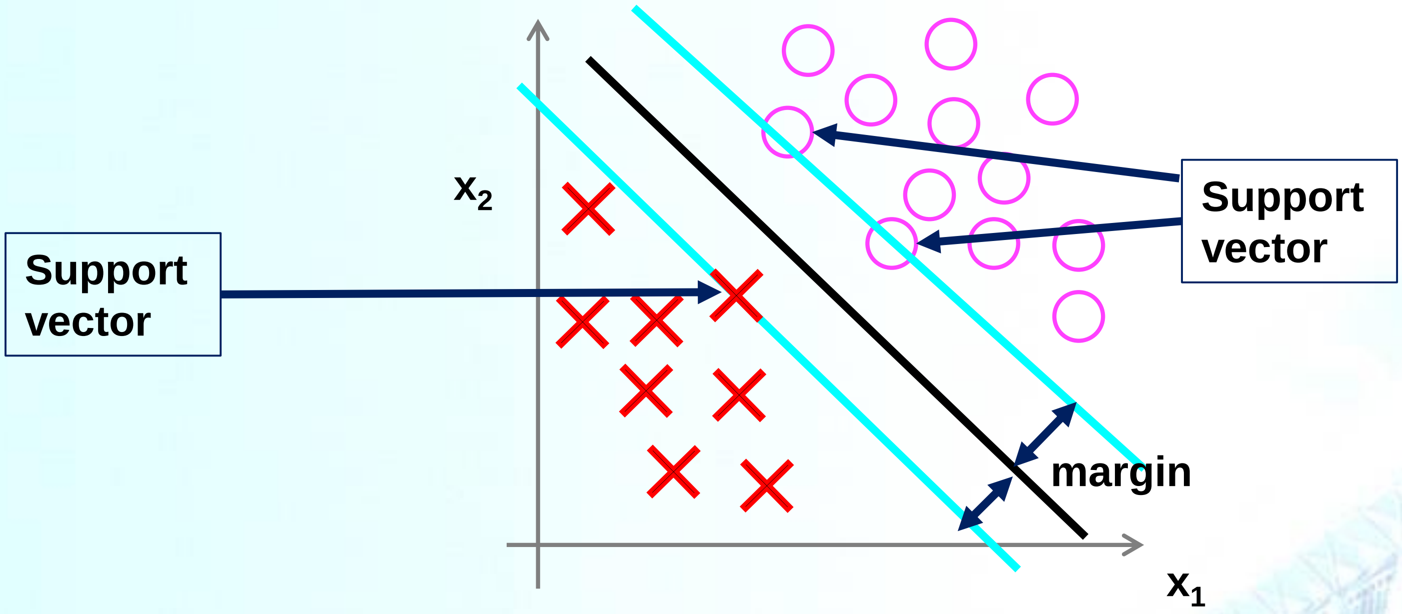
Support Vectors

- Moving a support vector
  - moves the decision boundary



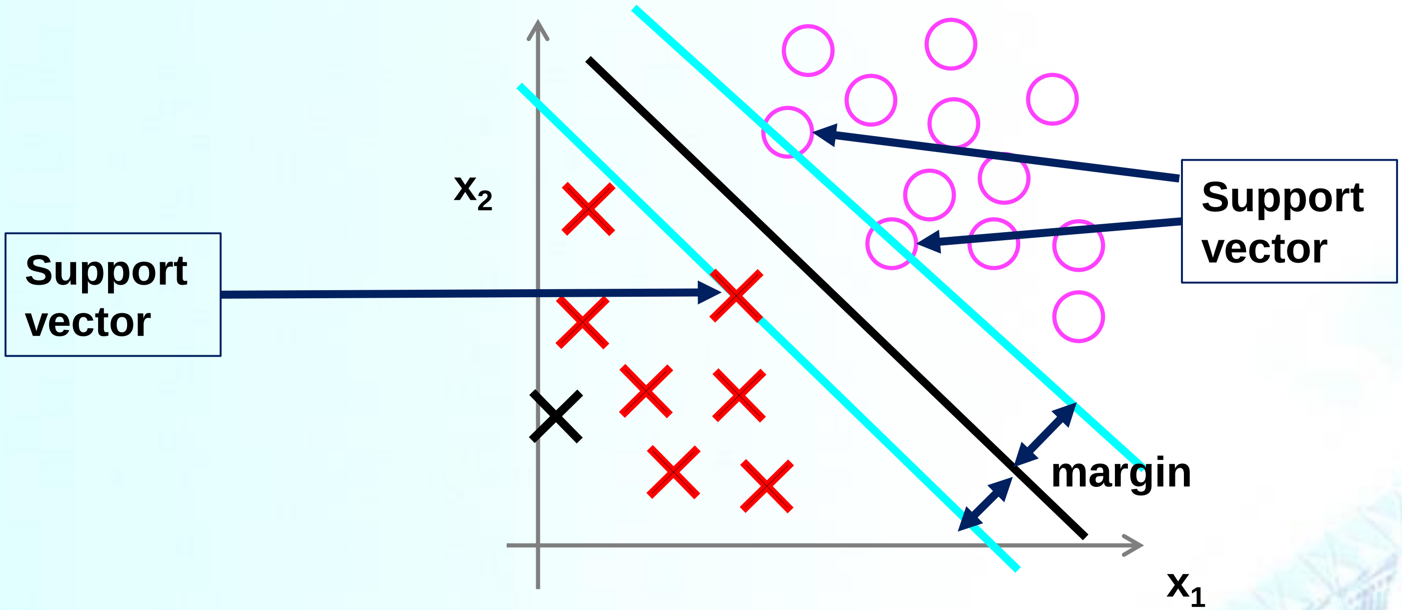
Support Vectors

- Moving the other vector
  - Does not have any effect



Support Vectors

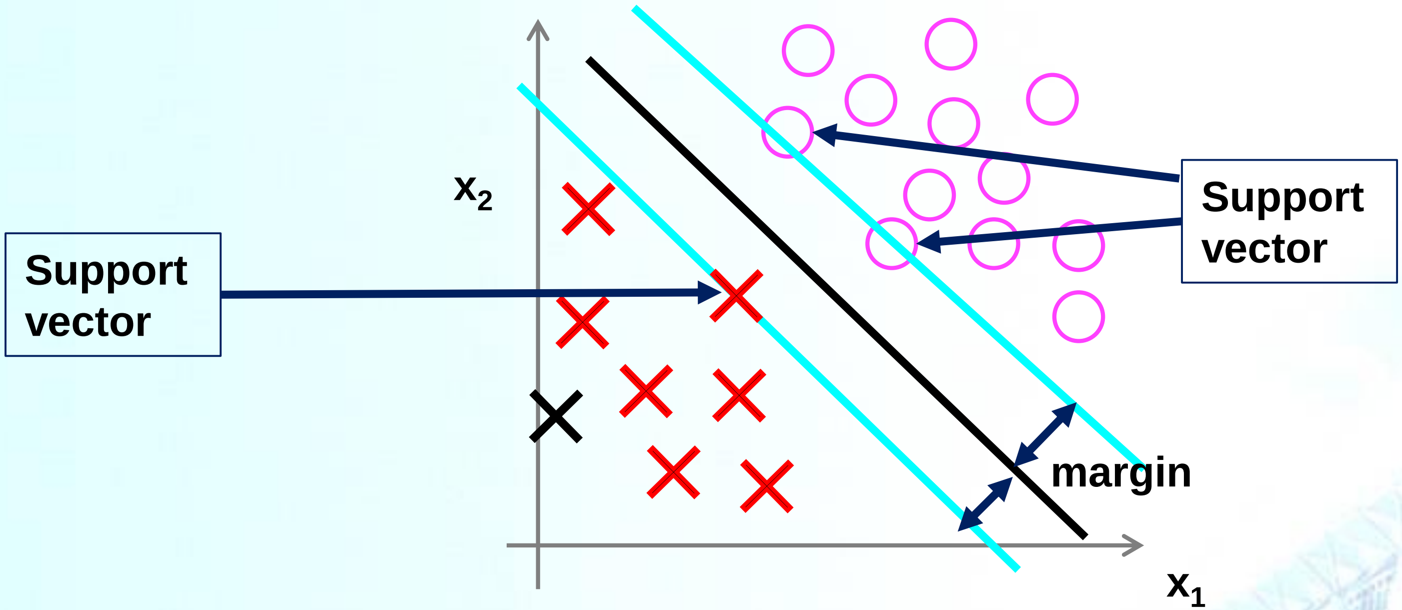
- Moving the other vector
  - Does not have any effect



Support Vectors

- Moving the other vector
  - Does not have any effect

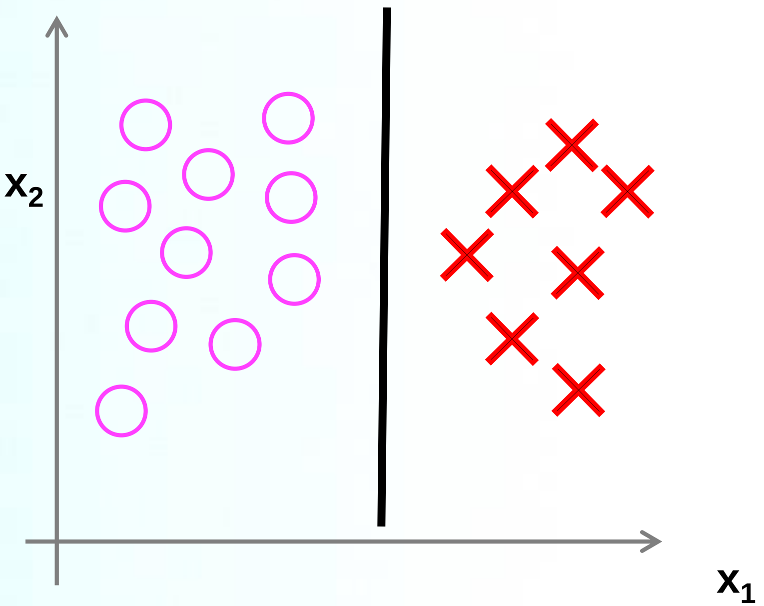
➔ Only the support vectors determine the weights and thus the boundary



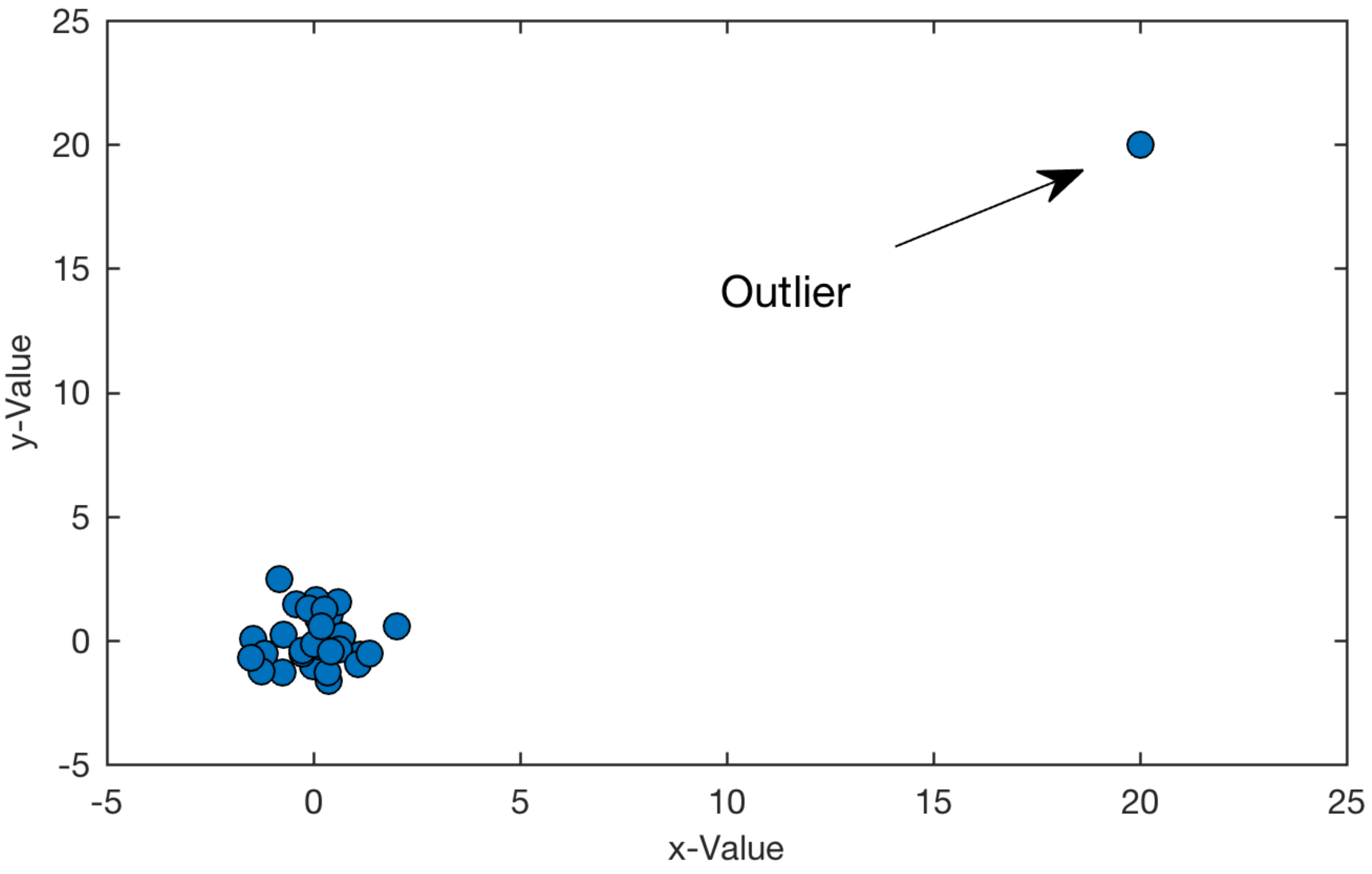
Embedded System Lab.

Large Margin Classifier without Outliers

- When C is very large,
  - Without outliers



Outlier



Outlier

- Property of one person
  - Two billion dollars
- Average property of 299 persons
  - 3 million dollars
- Average property of 300 persons
  - ?

Outlier

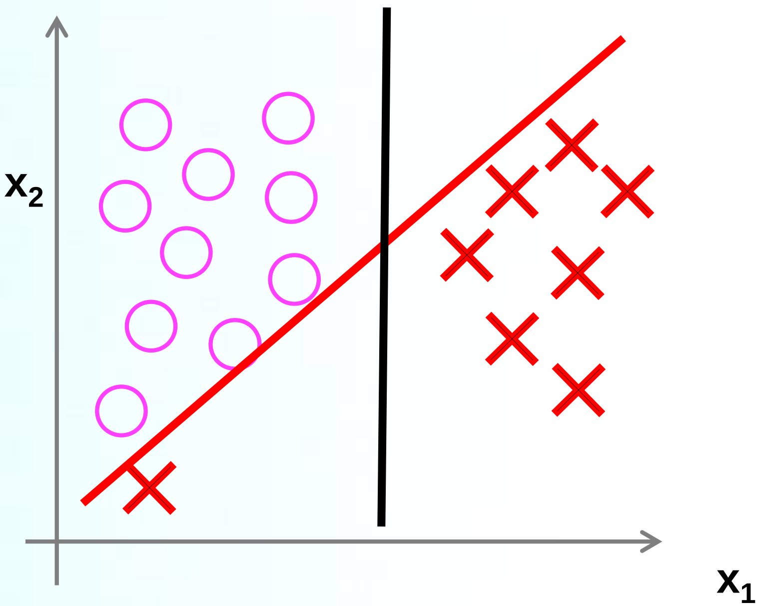
- Outlier (in statistics)
  - An observation point that is distant from other observations

- Outlier occurrence
  - Measurement error
    - One wishes to discard them or use statistics that are robust to outliers,
  - The population has a heavy-tailed distribution
    - they indicate that the distribution has high skewness
      - one should be very cautious in using tools or intuitions that assume a normal distribution

- Frequent cause of outliers
  - A mixture of two distributions
    - which may be two distinct sub-populations
    - or may indicate 'correct trial' versus 'measurement error'

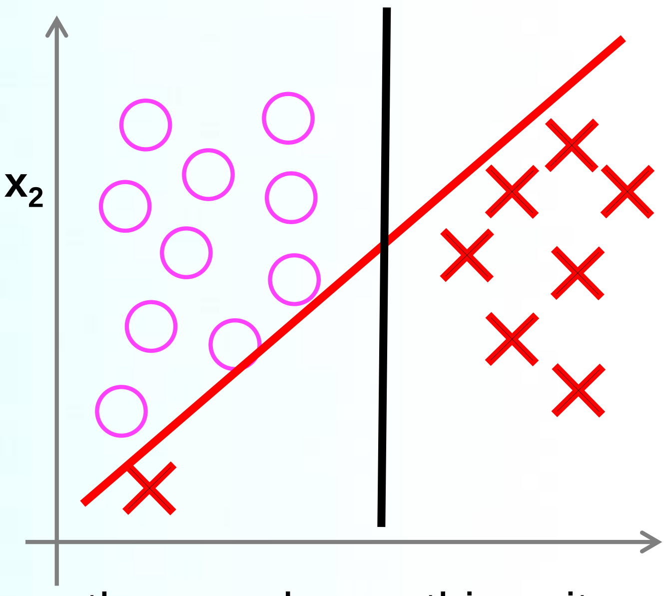
Large Margin Classifier in Presence of Outliers

- When C is very large,
  - SVM is very sensitive to outliers with using large margin ONLY



Large Margin Classifier in Presence of Outliers

- When C is very large,
  - SVM is very sensitive to outliers with using large margin ONLY
  - A single example might not represent a good reason to change an algorithm



- If C is very large, then we do use this quite naive max the margin approach
- But if C is reasonably small, or a not too large, then we stick with the black decision boundary

SVM

- What about non-linearly separable data?
  - Then SVM still does the right thing for a normal size C
- Idea of SVM of a large margin classifier
  - only really relevant with no outliers and linearly separable data

Embedded System Lab.

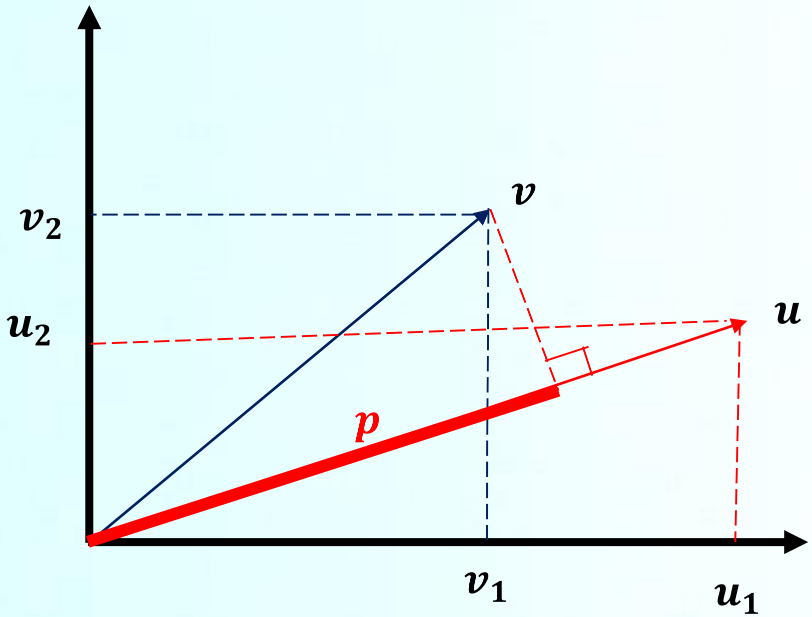
Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

Vector Inner Product

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{bmatrix} \qquad \boldsymbol{v} = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix}$$

Vector Inner Product

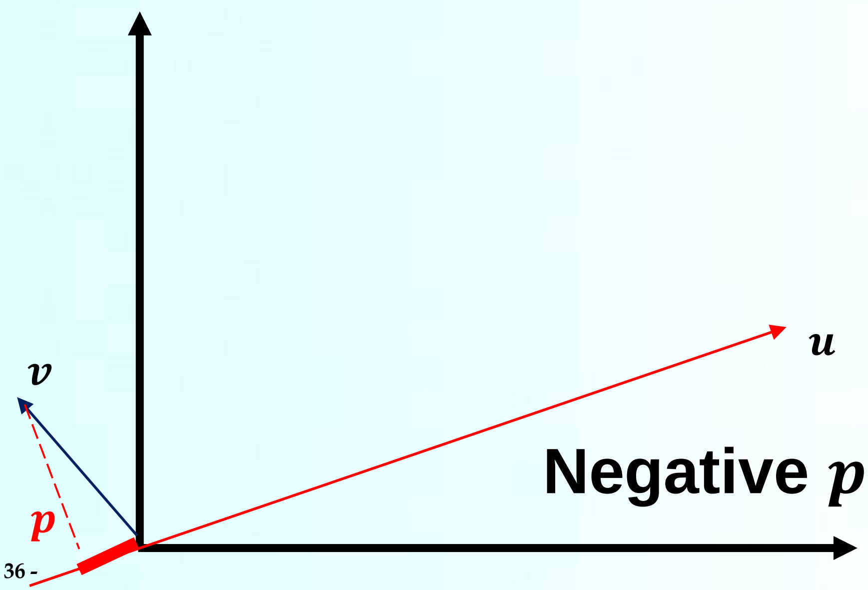


$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Norm of u = $\|u\| = \sqrt{u_1^2 + u_2^2}$

p: the length of projection of v onto u

$$u^T v = p\|u\| = u_1 v_1 + u_2 v_2$$
$$= v^T u$$



Negative p

SVM Decision Boundary

- SVM decision boundary

$$min_{\theta}\left(\frac{1}{2}\sum_{j=1}^{n}\theta_j^2\right)$$

$$\text{such that} \quad \begin{array}{lll} \theta^T x \geq 1 & \text{if} & y^{(i)} = 1 \\ \theta^T x \leq -1 & \text{if} & y^{(i)} = 0 \end{array}$$

- Assume $\theta_0 = 0$ and $n = 2$.
  - Then the SVM is minimizing the squared norm.

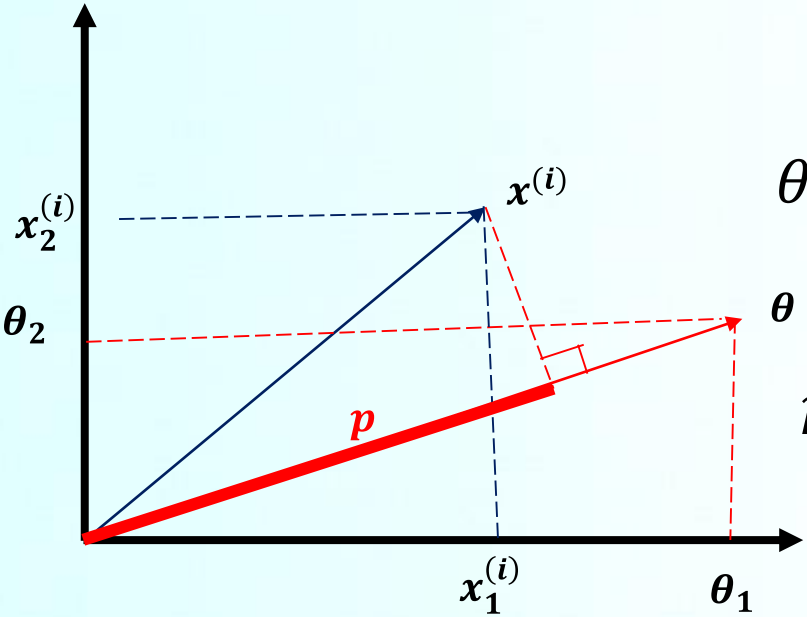
$$min_{\theta}\left(\frac{1}{2}||\theta||^2\right)$$

$$\text{such that} \quad \begin{array}{lll} \theta^T x \geq 1 & \text{if} & y^{(i)} = 1 \\ \theta^T x \leq -1 & \text{if} & y^{(i)} = 0 \end{array}$$

Embedded System Lab.

SVM Decision Boundary

SVM Decision Boundary



$$\theta^T x^{(i)} = p^{(i)} \|\theta\| = \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)}$$

$\boldsymbol{p}^{(i)}$: the projection of $\boldsymbol{x}^{(i)}$ onto the vector $\boldsymbol{\theta}$

- SVM

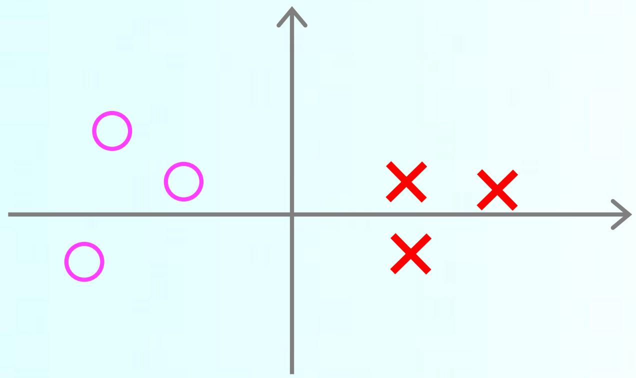
$$min_\theta \left(\frac{1}{2}\|\theta\|^2\right)$$

such that $p^{(i)}\|\theta\| \geq 1$ if $y^{(i)} = 1$

$p^{(i)}\|\theta\| \leq -1$ if $y^{(i)} = 0$

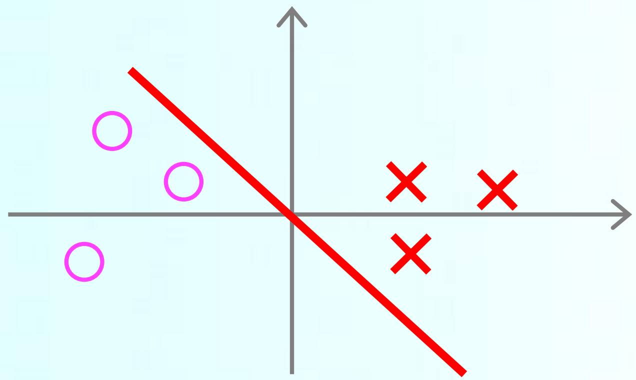
SVM Decision Boundary

- Given the following training examples,
  - What is the appropriate boundary?
- Assumption of $\theta_0 = 0$
  - The boundary has to pass through the origin (0,0)



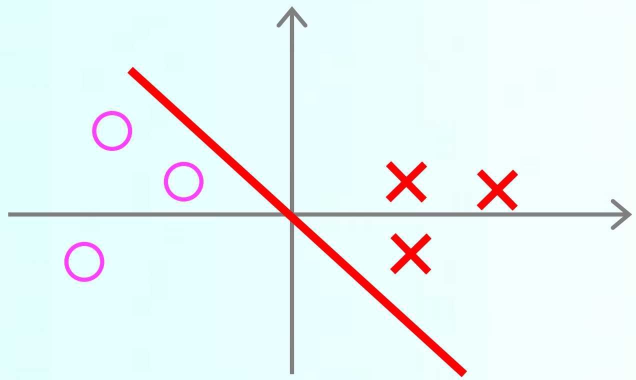
SVM Decision Boundary

- The following red line
  - Small margins
    - Decision boundary comes very clos to examples.
  - SVM does not choose the line



SVM Decision Boundary

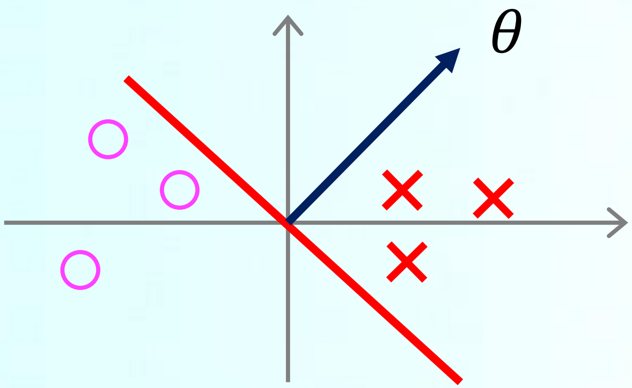
- The following red line
  - Small margins
    - Decision boundary comes very clos to examples.
  - SVM does not choose the line
    - why ?



Embedded System Lab.

SVM Decision Boundary

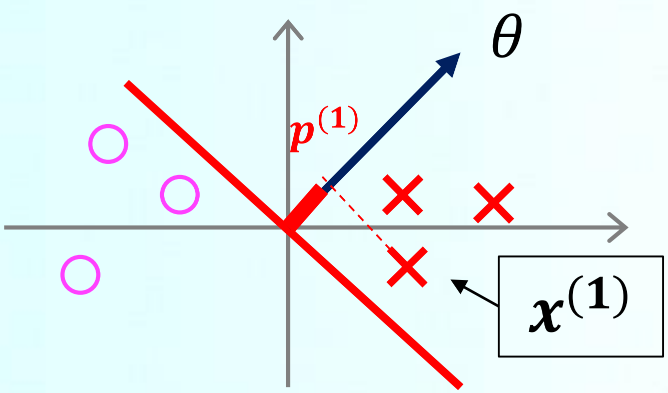
- θ is always at 90 degrees to the decision boundary.



SVM Decision Boundary

- θ is always at 90 degrees to the decision boundary.

- Select the first example.
  - Project a line from $\boldsymbol{x}^{(1)}$ onto the θ vector

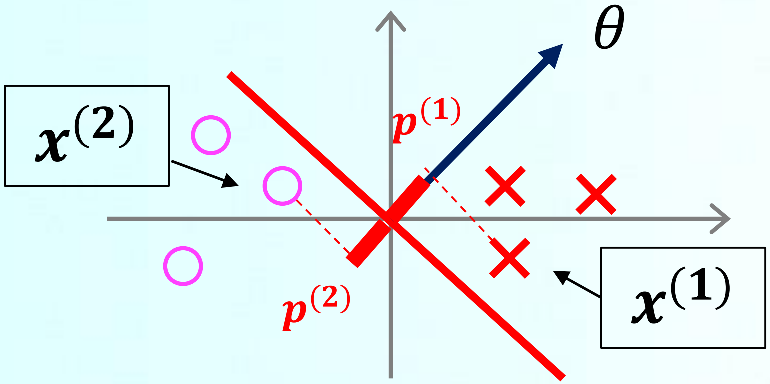


SVM Decision Boundary

- θ is always at 90 degrees to the decision boundary.

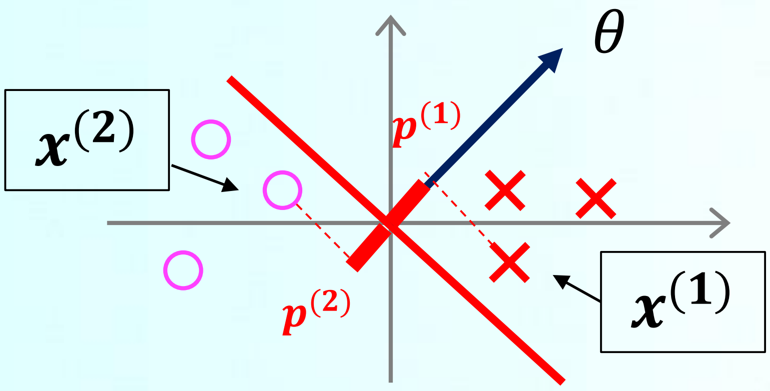
- Select the first example.
  - Project a line from $\boldsymbol{x}^{(1)}$ onto the θ vector

- Select the 2$^{\text{nd}}$ example.



SVM Decision Boundary

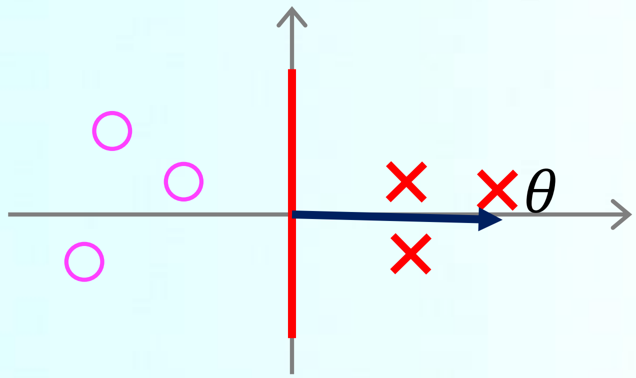
- Need $p^{(1)}||\theta|| \geq 1$ for positive examples
  - If $p^{(1)}$is small
    - Means that $||\theta||$ must be pretty large
- Need $p^{(2)}||\theta|| \leq -1$ for negative examples
  - If $p^{(2)}$is a small negative number
    - Means that $||\theta||$ must be pretty large



- Considering the optimization objective of finding a set of parameters where $||\theta||$ is small,
  - Not good parameter vector (because smaller p values bring larger $||\theta||$

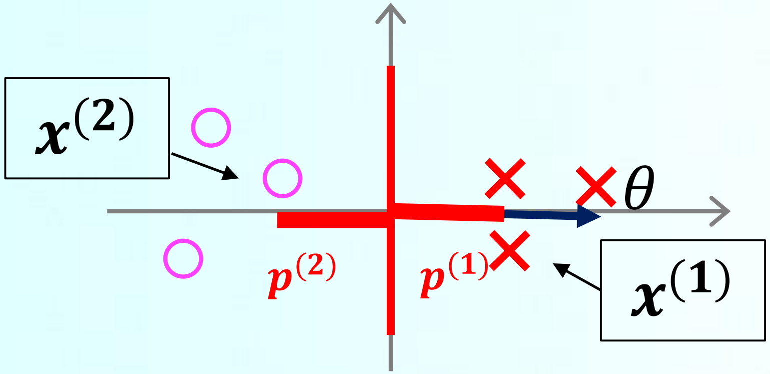
SVM Decision Boundary

- Choose another boundary



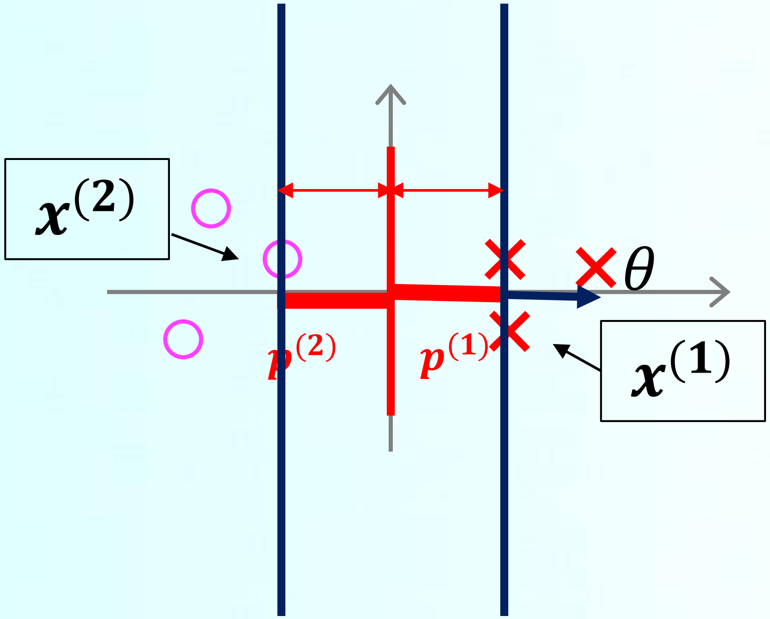
SVM Decision Boundary

- Choose another boundary
  - $p^{(1)}$ becomes larger
    - Means that $||\theta||$ can be small
  - $p^{(2)}$ becomes larger-magnitude and negative
    - Means that $||\theta||$ can be small
  - ➔ can make $||\theta||$ smaller
  - ➔ Which is why the SVM chooses this boundary as better



SVM Decision Boundary

- Large margin effect
  - Margin magnitude is a function of the p values
    - By maximizing these p values ➔ minimizing ||θ||



SVM Decision Boundary

- If this is the case, we are entertaining
  only decision boundaries which pass through (0,0)

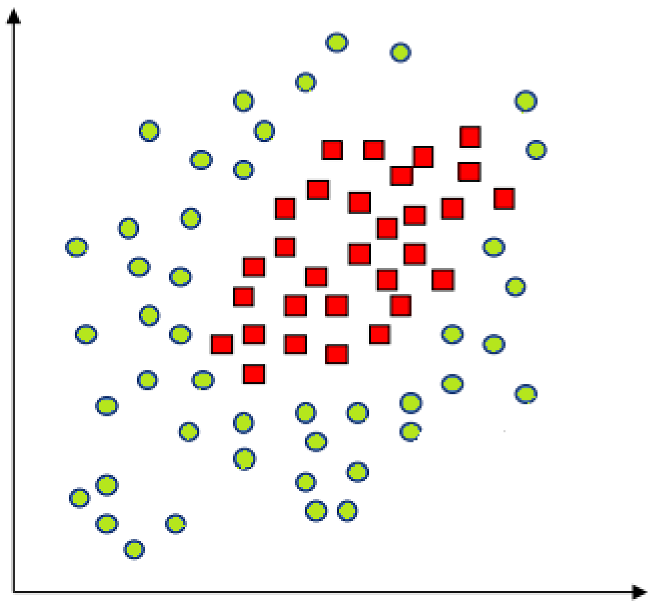
- For nonzero θ_0 ,
  - Decision boundaries cross through the x and y values at points other than (0,0)
    - Similar discussion for SVM (when C is very large)
      - SVM is looking for a large margin separator btw the classes

Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

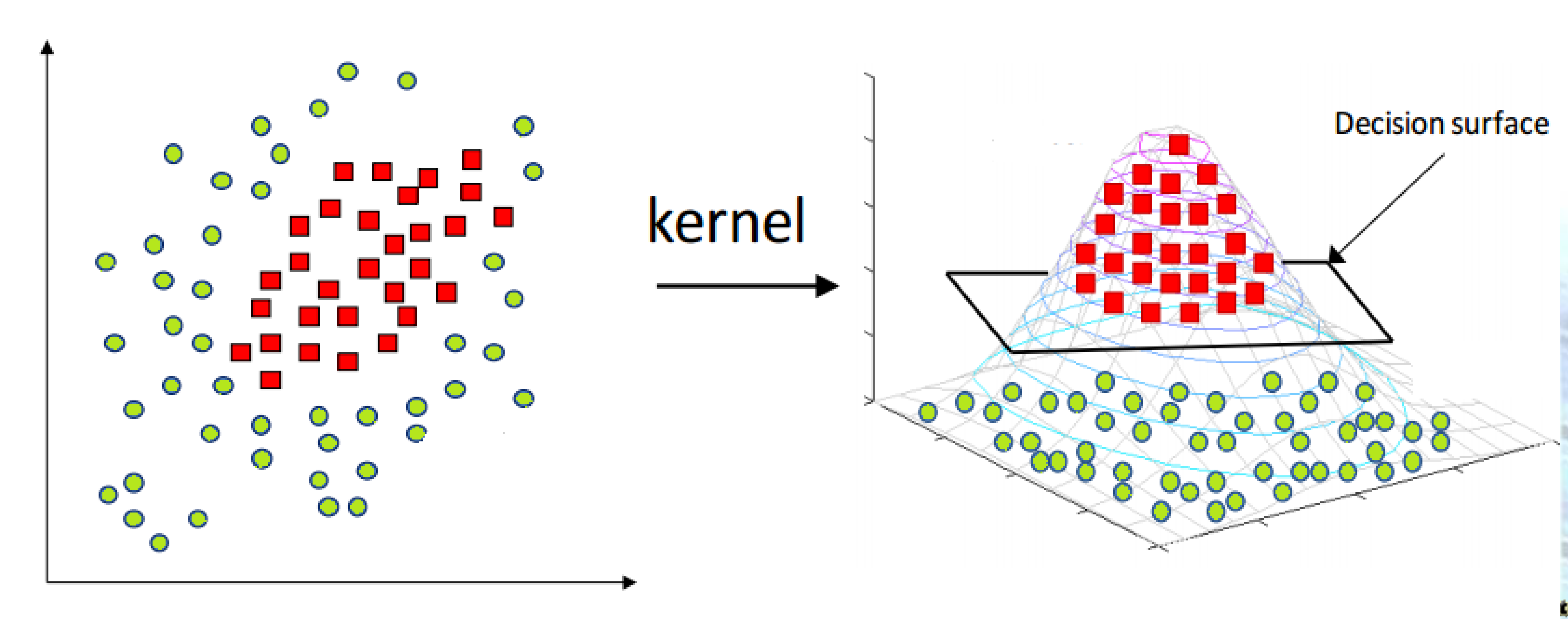
Nonlinear Decision Boundary

- Given a training set, we want to find a non-linear boundary



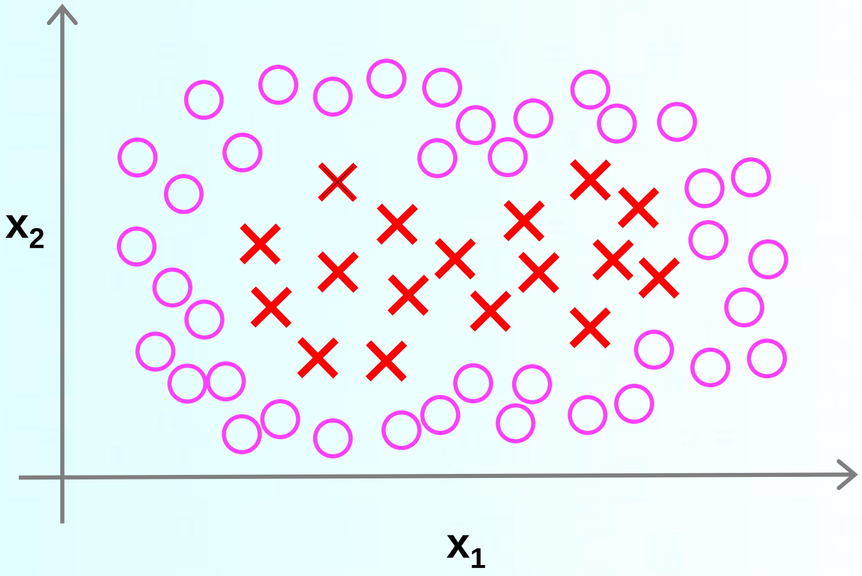
Nonlinear Decision Boundary

- Given a training set, we want to find a non-linear boundary
  - SVM can handle such situations by using a kernel function
    - which maps the data to a different space where a linear hyperplane can be used to separate classes
      - This is known as the **kernel trick** where the kernel function transforms the data into the higher dimensional feature space so that a linear separation is possible.



Nonlinear Decision Boundary

- Given a training set, we want to find a non-linear boundary

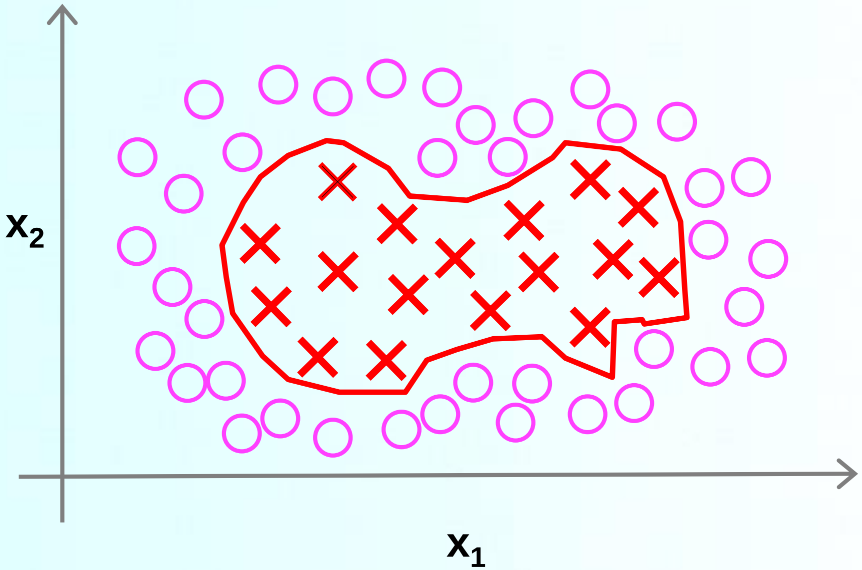


Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$

  - $h_\theta(x) = \begin{cases} 1 & \text{if} \quad \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \geq 0 \\ 0 & otherwise \end{cases}$



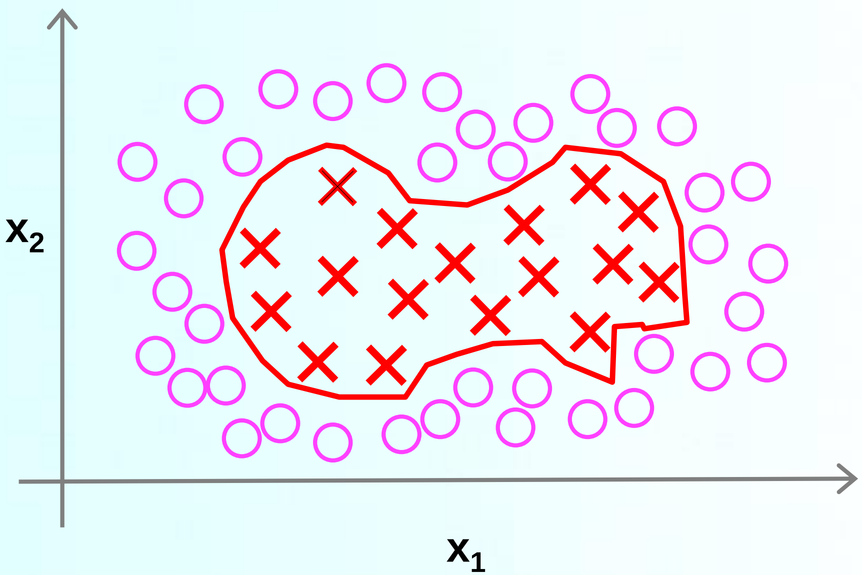
Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$

  - $h_\theta(x) = \begin{cases} 1 & \text{if} \quad \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \geq 0 \\ 0 & otherwise \end{cases}$
  - $h_\theta(x) = \theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \theta_4 f_4 + \theta_5 f_5$

    where $f_1 = x_1,\ f_2 = x_2, f_3 = x_1 x_2,\ f_4 = x_1^2,\ f_5 = x_2^2$



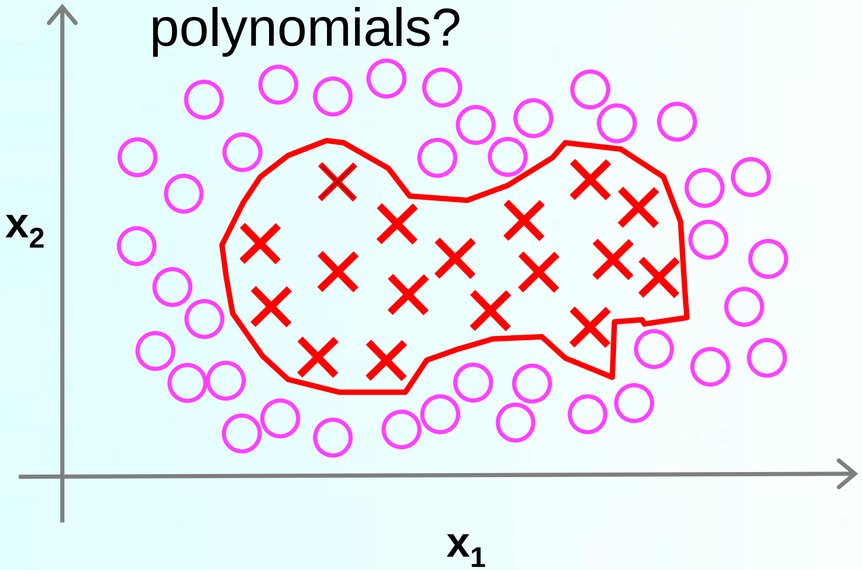
Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$

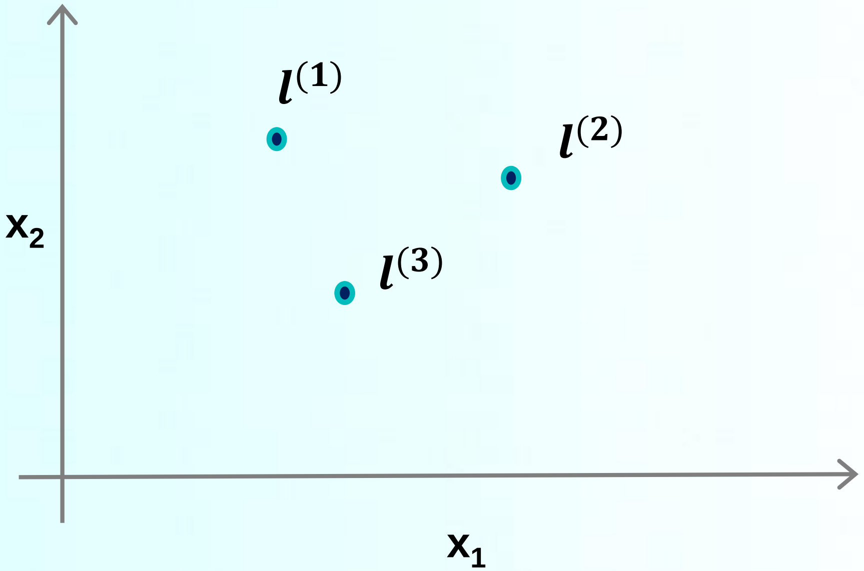
  - $h_\theta(x) = \begin{cases} 1 & \text{if} \quad \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \geq 0 \\ 0 & otherwise \end{cases}$
  - $h_\theta(x) = \theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \theta_4 f_4 + \theta_5 f_5$

    where $f_1 = x_1,\ f_2 = x_2, f_3 = x_1 x_2,\ f_4 = x_1^2,\ f_5 = x_2^2$
  - Is there a better choice of feature $f_1,\ f_2, f_3$ than the high order polynomials?



Kernel

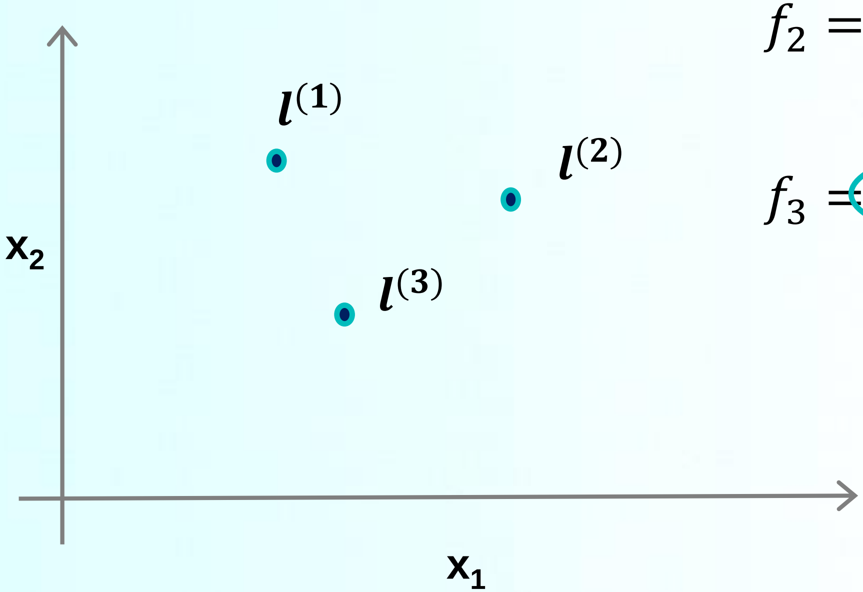
- New features
  - Define three features in this example (ignore x_0)
  - Have a graph of x_1 vs. x_2
  - Pick three points in that space
    - These points $\boldsymbol{l}^{(1)}$, $\boldsymbol{l}^{(2)}$, and $\boldsymbol{l}^{(3)}$, were chosen manually and are called **landmarks**
      - Given x, define f_1 as the similarity btw $(x,\ l^{(1)})$



Kernel

- Given x,

  compute new feature depending on proximity to landmarks $\boldsymbol{l}^{(1)}$, $\boldsymbol{l}^{(2)}$, and $\boldsymbol{l}^{(3)}$



$$f_1 = similarity(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$$f_2 = similarity(x, l^{(2)}) = \exp\left(-\frac{\|x - l^{(2)}\|^2}{2\sigma^2}\right)$$

$$f_3 = similarity(x, l^{(3)}) = \exp\left(-\frac{\|x - l^{(3)}\|^2}{2\sigma^2}\right)$$

kernel $\boldsymbol{k}(x, l^{(3)})$

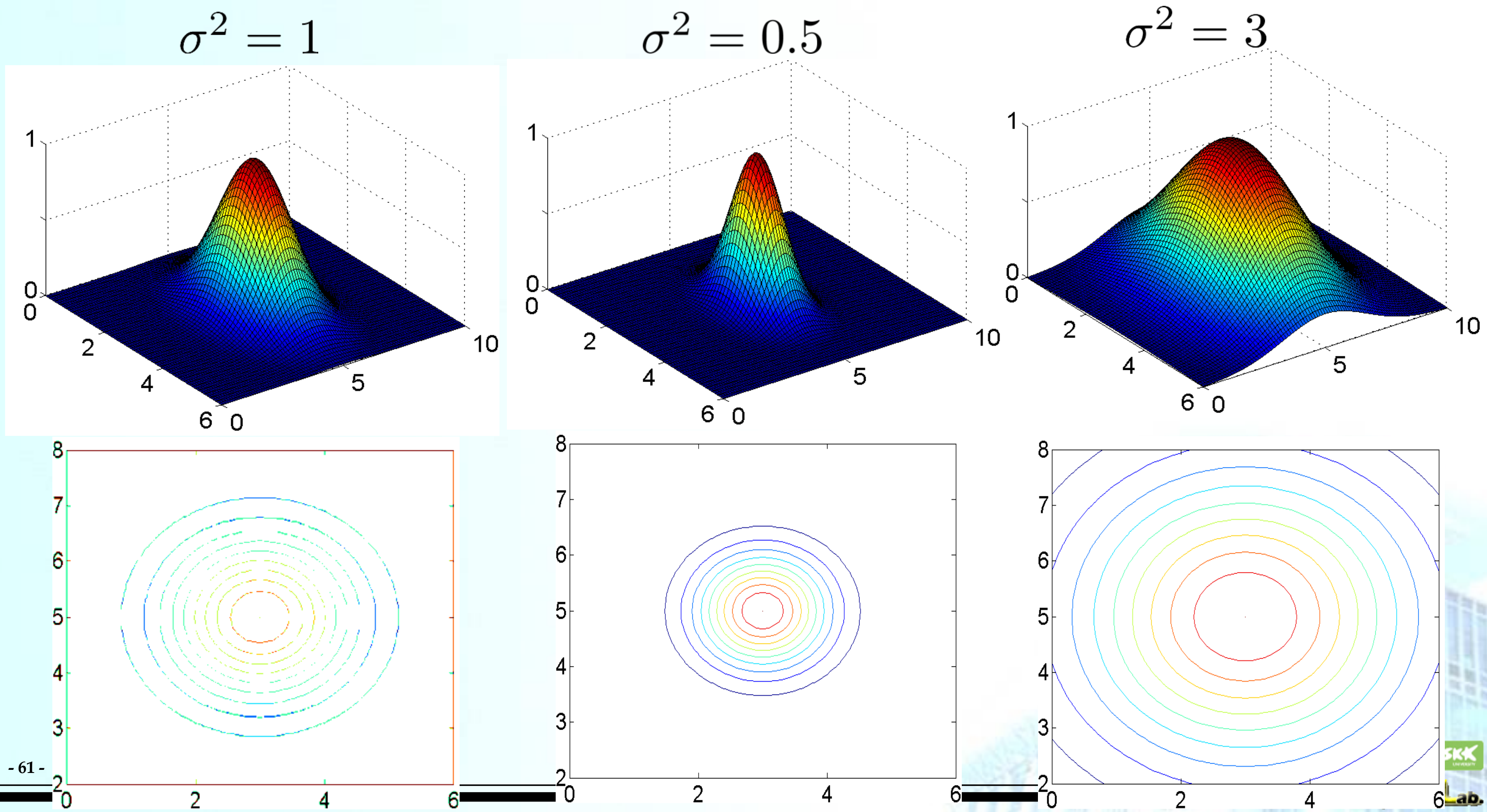
Gaussian kernel

Kernel and Similarity

- $f_1 = similarity\left(x, l^{(1)}\right) = \exp\left(-\frac{\left\|x-l^{(1)}\right\|^2}{2\sigma^2}\right)$
  - If $x \approx l^{(1)}$, $f_1 \approx \exp\left(-\frac{0^2}{2\sigma^2}\right) = 1$
  - If x is far from $l^{(1)}$, $f_1 = \exp\left(-\frac{(large\ num)^2}{2\sigma^2}\right) \approx 0$

Example

- $l = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\left\| x - l^{(1)} \right\|^2}{2\sigma^2}\right)$

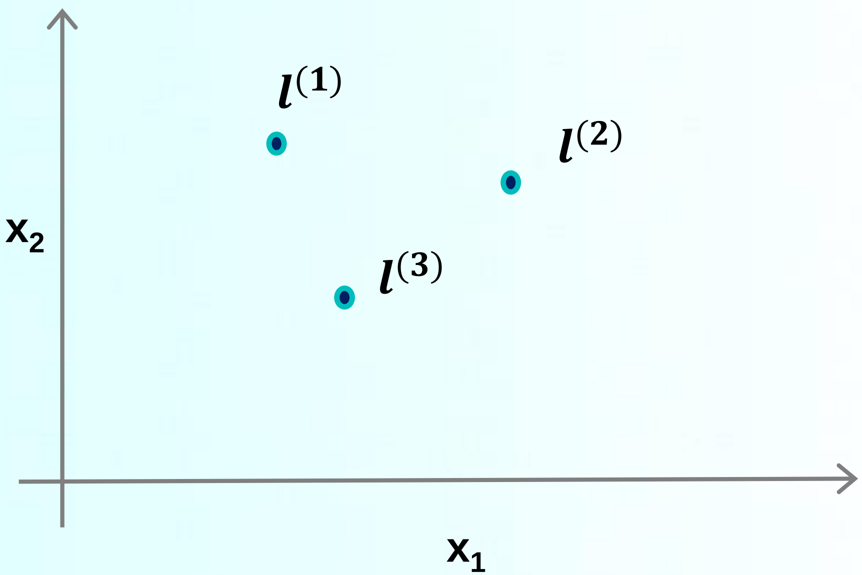


Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

  - Suppose that we get $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$ after training

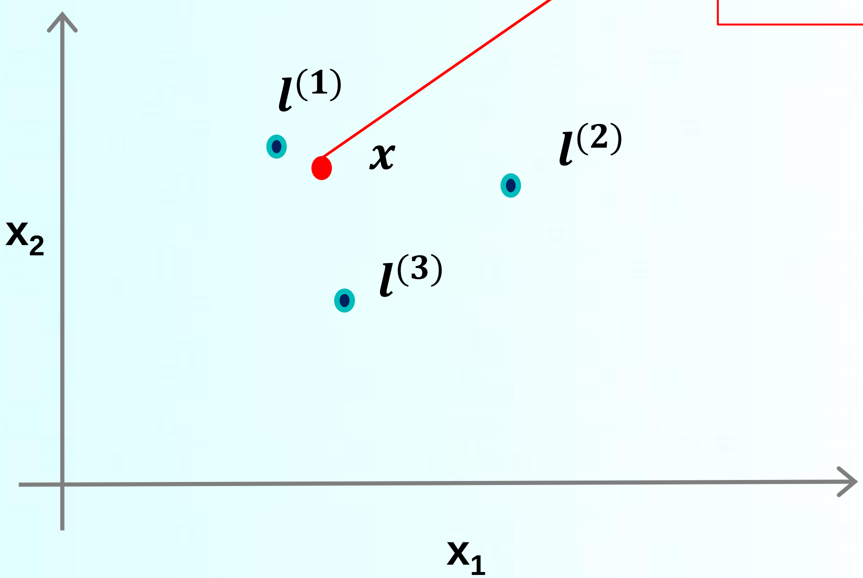


Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

  - Suppose that we get $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$ after training
    - Given the following x,



$f_1 \approx 1,\ f_2 \approx 0,\ f_3 \approx 0$

$-0.5 + 1 + 0 + 0 = 0.5 > 0$ ➔

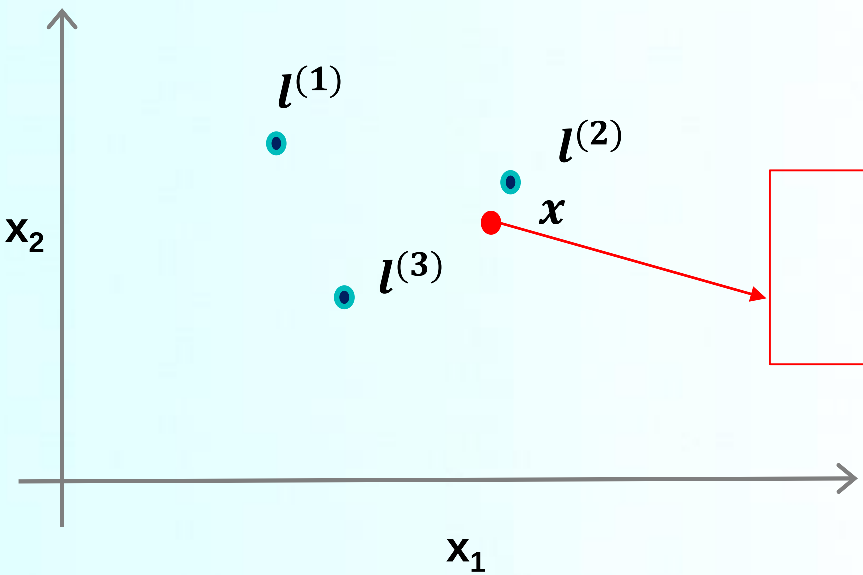
predict $y = 1$

Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

  - Suppose that we get $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$ after training
    - Given the following x,



$f_1 \approx 0,\ f_2 \approx 1,\ f_3 \approx 0$

$-0.5 + 0 + 1 + 0 = 0.5 > 0$ ➔

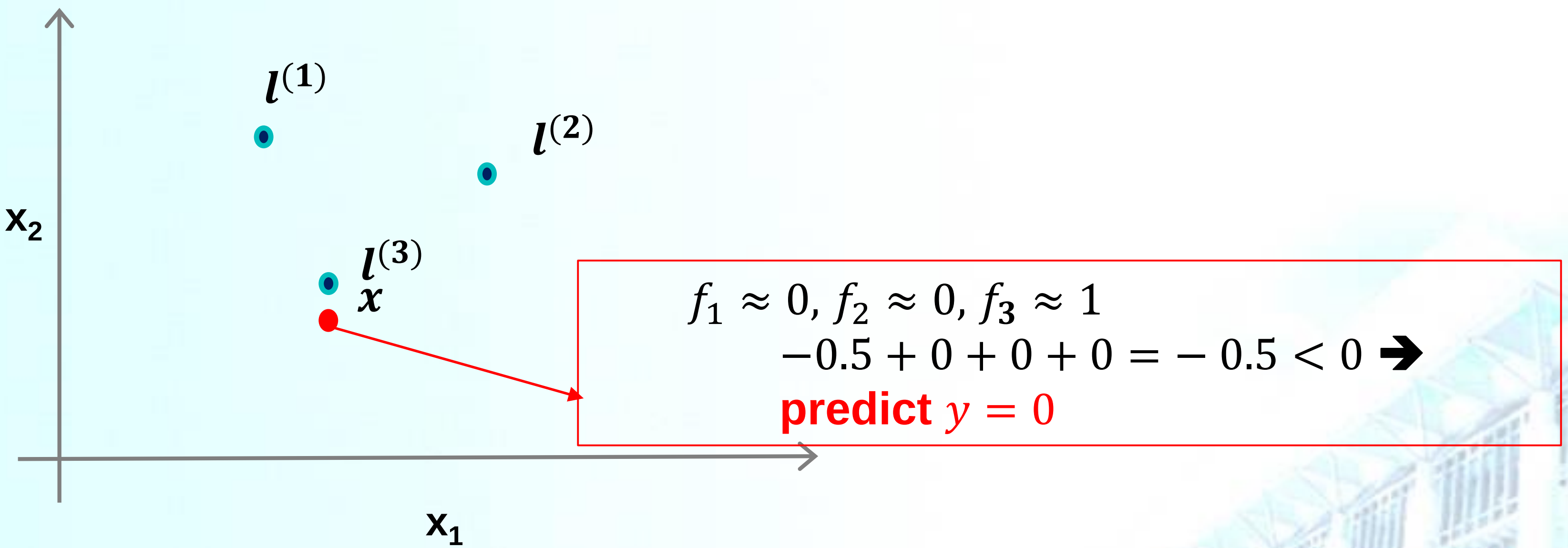
predict $y = 1$

Nonlinear Decision Boundary

- Predict $y = 1$

  if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

  - Suppose that we get $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$ after training
    - Given the following x,

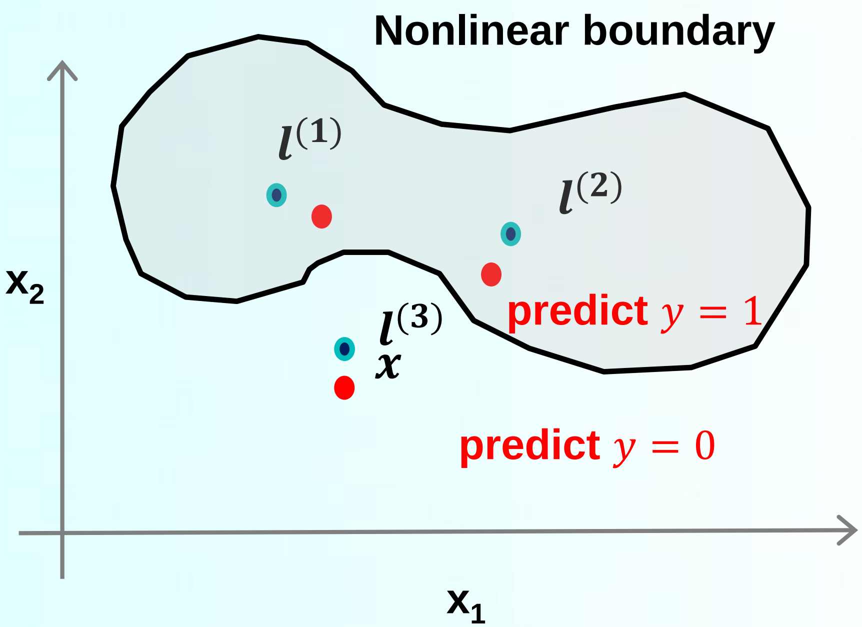


Nonlinear Decision Boundary

- Predict $y = 1$

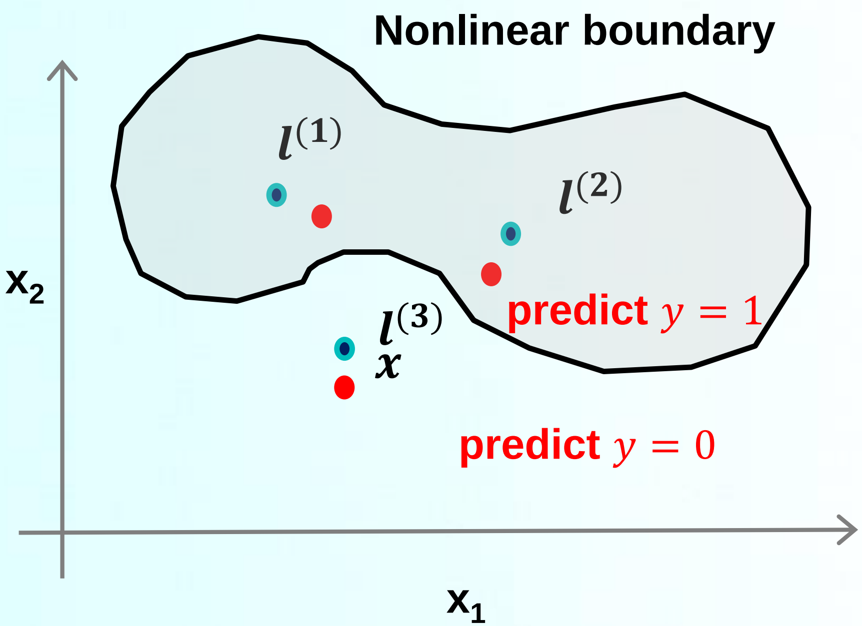
  if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

  - Suppose that we get $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$ after training
    - We may get the following a nonlinear decision boundary



Nonlinear Decision Boundary

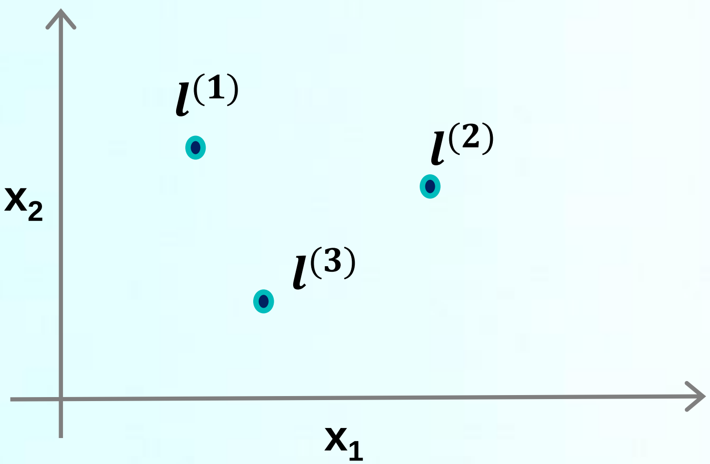
- How do we choose the landmarks?
- Can we use other kernels?



Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

Choosing The Landmarks

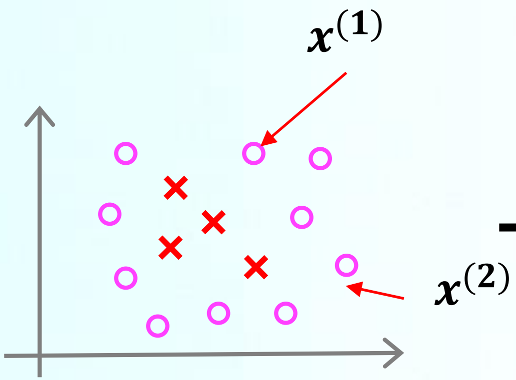


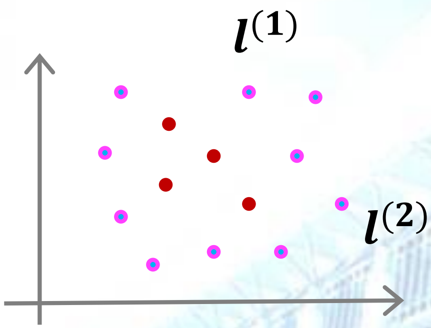
Given x :

$$f_1 = similarity(x, l^{(1)}) = \exp\left(-\frac{\left\|x - l^{(1)}\right\|^2}{2\sigma^2}\right)$$

Predict $y = 1$ if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Where to get $l^{(1)}, l^{(2)}, l^{(3)}, \ldots$?





SVM with Kernels

- Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$,
  - Choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \ldots, l^{(m)} = x^{(m)}$

- Given one example x,
  - $f_1 = similarity(x, l^{(1)})$
  - $f_2 = similarity(x, l^{(2)})$

  …
  - $f_m = similarity(x, l^{(m)})$

$$f = \begin{bmatrix} f_0 \\ f_1 \\ \cdots \\ f_m \end{bmatrix} \in R^{m+1}, \quad f_0 = 1$$

SVM with Kernels

- Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$,
  - Choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \ldots, l^{(m)} = x^{(m)}$

- For one training example $x^{(i)}$,
  - $f_1^{(i)} = similarity(x^{(i)}, l^{(1)})$
  - $f_2^{(i)} = similarity(x^{(i)}, l^{(2)})$

  …
  - $f_i^{(i)} = similarity(x^{(i)}, l^{(i)}) = \exp\left(-\frac{0}{2\sigma^2}\right) = 1$

  …
  - $f_m^{(i)} = similarity(x^{(i)}, l^{(m)})$

SVM with Kernels

- Hypothesis: Given x , compute features $f = \begin{bmatrix} f_0 \\ f_1 \\ \cdots \\ f_m \end{bmatrix} \in R^{m+1}, \quad f_0 = 1$
  - Predict $y = 1 \;\; if \;\; \theta^T f \geq 0,$ (where $\theta \in R^{m+1}$ and $f \in R^{m+1}$)
    - How do we get θ ?

- SVM learning algorithm

$$min_\theta C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T f^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T f^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{\cancel{n}\; \boldsymbol{m}} \theta_j^2$$

  - If we ignore θ_0, $\sum_{j=1}^{n} \theta_j^2 = \theta^T \theta$

SVM with Kernels

- Many implementations use

$$\theta^T M \theta$$

where the matrix *M* depends on the kernel they use

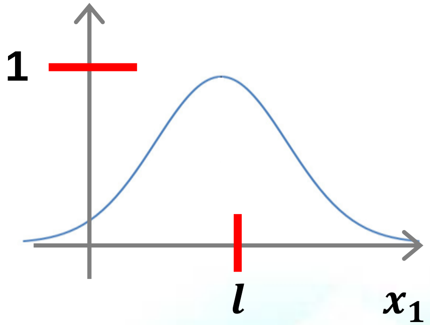
  - Gives a slightly different minimization
    - It means we determine a rescaled version of θ
  - Allows more efficient computation, and scale to much bigger training sets
  - If there is a training set with 10,000 values,

    it means there are 10,000 features
    - It can be expensive to solve for all these parameters
    - So by adding this,
      - we avoid <u>one `for` loop</u> and use <u>one matrix multiplication</u> algorithm instead

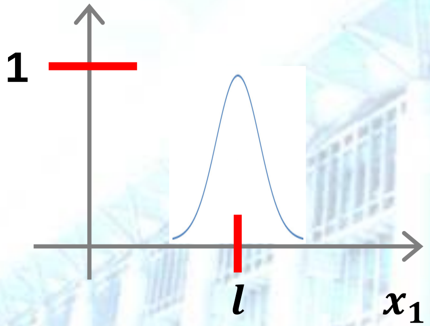
SVM Parameters

- $C = \frac{1}{\lambda}$
  - Large C
    - Lower bias, high variance
  - Small C
    - Higher bias, low variance

- Large σ^2
  - Features f_i vary more smoothly.
  - Higher bias, lower variance.



- Small σ^2
  - Features f_i vary less smoothly.
  - Lower bias, higher variance.



Outline

- Optimization Objective
- Large margin intuition
- Mathematics behind large margin classification
- Kernel I
- Kernel II
- Using an SVM

SVM Implementation

- SVM implementation
  - Use SVM software packages (e.g. **`liblinear, libsvm`**) to solve parameters θ
  - Need to specify
    - Choice of parameter C
    - Choice of kernel (similarity function)
  - Example
    - No kernel
      - "linear kernel"
    - Gaussian kernel
      - $f_i = \exp\left(-\frac{\left\|x - l^{(i)}\right\|^2}{2\sigma^2}\right)$, where $l^{(i)} = x^{(i)}$

SVM Implementation

- No kernel - **linear kernel**
  - Predict $y = 1 \ \ if \ \ \theta^T x \geq 0$
    - So no f vector
    - Get a standard linear classifier
  - Why do this?
    - For large n (lots of features) and small m (few examples)
      - Not enough data
        - risk overfitting in a high dimensional feature-space

SVM Implementation

- **Gaussian kernel**
  - $f_i = \exp\left(-\frac{\left\|x - l^{(i)}\right\|^2}{2\sigma^2}\right)$, where $l^{(i)} = x^{(i)}$
    - Need to choose σ^2
  - When would we choose a Gaussian?
    - For small n (few features) and large m (lots of examples)
      - e.g. 2D training set that is large

SVM Implementation

Gaussian kernel

- When using a Gaussian kernel, implement the kernel function

```
function f = kernel(x1,x2)
```

$$f = exp\left(\frac{\|x1 - x2\|^2}{2\sigma^2}\right)$$

```
return
```

  - f returns a real number
  - Some SVM packages will expect you to define kernel
    - Some SVM implementations include the Gaussian and a few others
      - Gaussian is probably most popular kernel

- Make sure you perform **feature scaling** before using a Gaussian kernel
  - Otherwise, features with a large value will dominate the f value
    - $\|x - l\|^2 = \underbrace{(x_1 - l_1)^2}_{\text{1000 feet}^2} + \underbrace{(x_2 - l_2)^2}_{\text{1-5 bedrooms}} + \cdots + (x_n - l_n)^2$

SVM Implementation

- Other choice of kernel
  - Linear and Gaussian are most common
  - Not all similarity functions make valid kernels
    - Must satisfy **Mercer's Theorem**
      - to make sure SVM packages' optimizations run correctly, and do not diverge
    - SVM use numerical optimization tricks
      - Mean certain optimizations can be made, but they must follow the theorem

Mercer's Theorem

- $K\colon [a,b]^2 \to R$
  - A symmetric, non-negative definite, continuous function.
- Then there exists
  - a countable sequence of functions $\{\emptyset_i\}_{i \in N}$ and
  - a sequence of positive real numbers $\{\lambda_i\}_{i \in N}$ such that,

$$K(s,t) = \sum_{i=1}^{\infty} \lambda_i \emptyset_i(s) \emptyset_i(t)$$

Mercer's Condition

- A real-valued function $K(x, y)$ fulfills Mercer's condition

  if for all square-integrable functions $g(x)$ one has

$$\int\int g(x)K(x,y)g(y)dxdy \geq 0$$

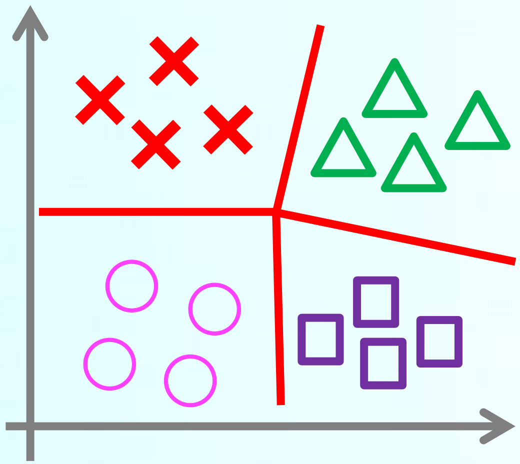
SVM Implementation

- Other choice of kernel
  - **Polynomial Kernel**
    - Measure the similarity of x and l by doing one of
      - $(x^T l)^2, (x^T l)^3$, $(x^T l + 1)^3$, $(x^T l + 5)^4$, …
      - General form: $(x^T l + Const)^D$
      - ➔ If they are similar, then the inner product tends to be large
    - Not used that often
    - Two parameters
      - Degree of polynomial (D)
      - Number you add to ($Const$)
    - Usually performs worse than the Gaussian kernel
    - Used when x and l are both non-negative

SVM Implementation

- Other choice of kernel
  - **String kernel**
    - Used if input is text strings
    - Use for text classification
  - **Chi-squared kernel**
  - **Histogram intersection kernel**

Multi-Class Classification



$y \in \{1,2,\dots,K\}$

- Many SVM packages have built-in multi-class classification functionality

- Otherwise, use one-vs all method
  - (Train K SVMs, one to distinguish $y = i$ from the rest, for $i = 1,2,\dots,K$),
  - Get $\theta^{(1)}, \theta^{(2)}, \cdots, \theta^{(K)}$ $\boldsymbol{y = K}$

$\boldsymbol{y = 1}$

    - Pick class i with largest $\left(\theta^{(i)}\right)^T x$

$\boldsymbol{y = 2}$

Logistic Regression vs. SVM

- n: # features ($x \in R^{n+1}$), m: # of training examples
  - If n is large (relative to m),
    - e.g. text classification problem
      - Feature vector dimension is 10,000
      - Training set is 10 – 1,000
    - Use logistic regression, or SVM without a kernel(“linear kernel”)
  - If n is small and m is intermediate
    - $n = 1 \sim 1{,}000, \;\; m = 10 \sim 10{,}000$
    - Use SVM with Gaussian kernel
  - If n is small and m is large
    - $n = 1 \sim 1{,}000, \;\; m \geq 50{,}000$
    - Create/add more features,
      then use logistic regression or SVM without a kernel

Neural network likely to work well for most of these settings, but may be slower to train.

Logistic Regression vs. SVM

- Logistic regression and SVM with a linear kernel are pretty similar
  - Do similar things
  - Get similar performance
- A lot of SVM's power is using different kernels to learn complex non-linear functions
- SVM has a convex optimization problem
  - We get a global minimum
- SVM is widely perceived a very powerful learning algorithm

References

- https://www.coursera.org/learn/machine-learning
- http://www.holehouse.org/mlclass/12_Support_Vector_Machines.html
- R. Berwick, An Idiot's guide to Support vector machines (SVMs)
- https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/